KLASIFIKASI RISIKO GAGAL GINJAL KRONIS MENGGUNAKAN *EXTREME LEARNING MACHINE*

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Dimas Prenky Dicky Irawan
NIM: 145150200111002

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2018

PENGESAHAN

KLASIFIKASI RISIKO GAGAL GINJAL KRONIS MENGGUNAKAN *EXTREME LEARNING MACHINE*

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Dimas Prenky Dicky Irawan
NIM: 145150200111002

Skripsi ini telah diuji dan dinyatakan lulus pada
29 Juni 2018
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Imam Cholissodin, S.Si, M.Kom	Edy Santoso, S.Si, M.Kom
NIK: 201201 850719 1 001	NIP: 19740414 200312 1 004

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T, M.T, Ph.D
NIP: 19710518 200312 1 001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 29 Juni 2018

Dimas Prenky Dicky Irawan

NIM: 145150200111002

KATA PENGANTAR

Puji dan syukur penulis panjatkan kehadirat Allah SWT atas anugerah serta limpahan rahmat-Nya sehingga penulis dapat menyelesaikan skripsi dengan judul "Klasifikasi Risiko Gagal Ginjal Kronis Menggunakan *Extreme Learning Machine*" ini. Skripsi ini disusun sebagai syarat memperoleh gelar sarjana pada Program Studi Informatika/ Ilmu Komputer, Fakultas Ilmu Komputer Universitas Brawijaya.

Dalam proses penyelesaian skripsi ini penulis mendapatkan banyak bantuan, baik bantuan moral maupun materiil dari berbagai pihak. Oleh karena itu, penulis mengucapkan banyak terimakasih kepada:

1. Bapak Imam Cholissodin, S.Si, M.Kom selaku Pembimbing I yang telah memberikan bimbingan, arahan, ilmu, dan masukan dalam penyelesaian skripsi ini.
2. Bapak Edy Santoso, S.Si, M.Kom, selaku Pembimbing II yang juga telah memberikan bimbingan, arahan, ilmu, dan masukan dalam penyelesaian skripsi ini.
3. dr. Nur Samsu, Sp.PD-KGH selaku pakar yang telah memberikan penjelasan terkait Gagal Ginjal Kronis.
4. Bapak Wayan Firdaus Mahmudy, S.Si, M.T, Ph.D selaku Dekan Fakultas Ilmu Komputer Universitas Brawijaya.
5. Bapak Tri Astoto Kurniawan, S.T, M.T, Ph.D selaku Ketua Jurusan Teknik Informatika Fakultas Ilmu Komputer Universitas Brawijaya.
6. Bapak Agus Wahyu Widodo, S.T, M.Cs selaku Ketua Program Studi Teknik Informatika Fakultas Ilmu Komputer Universitas Brawijaya.
7. Orang tua penulis, Bapak Nani dan Ibu Tumpiati Choir'yah yang terus memberikan motivasi, semangat, serta dukungan moral maupun financial serta do'a dan juga adikku Yoga yang memberikan motivasi, semangat, kelucuan yang ihklas lillahi ta ala.
8. Seluruh civitas akademika Informatika Universitas Brawijaya yang telah banyak memberi bantuan dan dukungan selama penulis menempuh studi di Informatika Universitas Brawijaya dan selama penyelesaian skripsi ini.
9. Teman – teman Program Studi Informatika/Ilmu Komputer yang telah banyak sekali membantu dalam menempuh perkuliahan, masukan dan semangat yang diberikan kepada penulis sehingga terselesaikannya skripsi ini.
10. Teman-teman kontrakan Merjosari no 7 yang selalu memberi semangat, motivasi, kekuatan, kebersamaan dan dukungan dalam penyelesaian skripsi ini.

11. Semua pihak yang tidak dapat disebutkan satu per satu, yang telah membantu dan terlibat baik secara langsung maupun tidak langsung dalam penulisan skripsi ini.

Penulis menyadari bahwa skripsi ini tidak lepas dari kesalahan dan kekurangan. Oleh karena itu, penulis bersedia menerima kritik dan saran yang membangun untuk memperbaiki diri. Penulis berharap semoga skripsi ini dapat memberi manfaat.

Malang, 29 Juni 2018

Penulis
prenkydimas@gmail.com

ABSTRAK

Dimas Prenky Dicky Irawan, Klasifikasi Risiko Gagal Ginjal Kronis Menggunakan *Extreme Learning Machine*

Pembimbing: Imam Cholissodin, S.Si, M.Kom dan Edy Santoso, S.Si, M.Kom

Ginjal merupakan sebuah organ pada manusia yang mempunyai peranan sangat penting dalam proses mengatur kebutuhan cairan dan elektrolit. Gagal ginjal kronis merupakan sebuah penyakit terhadap ginjal yang terjadi karena infeksi ginjal serta adanya sumbatan yang dikarenakan batu ginjal. Untuk melakukan klasifikasi gagal ginjal kronis tenaga medis masih belum maksimal dalam menanganinya, untuk menangani masalah ini peneliti menggunakan *Extreme Learning Machine* untuk melakukan klasifikasi gagal ginjal kronis. *Extreme Learning Machine* merupakan sebuah algoritme klasifikasi yang mana algoritme ini merupakan bagian dari jaringan saraf tiruan yang memiliki *learning speed* bagus dan juga menurut penelitian yang sudah ada dihasilkan nilai akurasi yang begitu baik apabila dibandingkan menggunakan algoritme lainnya. Penelitian yang dilakukan ini mendapatkan perbandingan nilai data latih serta data uji optimal dengan nilai rasio 70:30, banyak *neuron hidden layer* sebesar 10 dan menggunakan fungsi aktivasi sigmoid bipolar dari parameter tersebut menghasilkan nilai akurasi sebesar 99,13%. Dari hasil akurasi yang didapatkan, menunjukkan bahwa metode *Extreme Learning Machine* cukup baik dipakai untuk proses klasifikasi gagal ginjal kronis.

Kata Kunci : Ginjal, Gagal Ginjal Kronis, *Extreme Learning Machine*, Jaringan Saraf Tiruan.

ABSTRACT

Dimas Prenky Dicky Irawan, Klasifikasi Risiko Gagal Ginjal Kronis Menggunakan *Extreme Learning Machine*

Pembimbing: Imam Cholissodin, S.Si, M.Kom dan Edy Santoso, S.Si, M.Kom

Kidney is an organ in humans that have a very important role in the process of managing fluid and electrolyte needs. Chronic renal failure is a disease of kidney that occurs due to kidney infection and the existence of blockage due to kidney stones. To perform the classification of chronic renal failure medical personnel are still not maximally in handling it, to deal with this problem researchers use the Extreme Learning Machine to perform the classification of chronic renal failure. The Extreme Learning Machine is a classification algorithm in which this algorithm is part of a neural network that has a good learning speed and also according to existing research results in a good accuracy value when compared to using other algorithms. This study obtained a comparison of the value of training data as well as the optimal test data with a 70:30 ratio value, many hidden layer neurons of 10 and using the bipolar sigmoid activation function of these parameters resulted in an accuracy of 99.13%. From the results of accuracy obtained, indicating that the method of Extreme Learning Machine is good enough to be used for the process of classification of chronic renal failure.

Keywords: *Kidney, Chronic Renal Failure, Extreme Learning Machine, Artificial Neural Network.*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR PERSAMAAN

DAFTAR KODE PROGRAM

LAMPIRAN

BAB 1 PENDAHULUAN

1.1 Latar belakang

Ginjal adalah organ vital yang mempunyai peranan sangat besar dalam proses pengaturan kebutuhan cairan serta elektrolit. Fungsi dari ginjal itu sendiri untuk mengatur air, mengatur konsentrasi jumlah garam pada darah, mengatur keseimbangan antara asam basa pada darah dan mengatur proses pembuangan zat-zat yang berbahaya bagi tubuh (Andriyani, 2015). Walaupun ginjal mempunyai peran besar untuk menjaga keseimbangan tubuh, bukan berarti ginjal akan terus bekerja secara maksimal. Seseorang apabila tidak menjaga kesehatan ginjal dengan baik tidak menutup kemungkinan kesehatan ginjalnya akan terganggu. Salah satu penyakit yang akan terjadi apabila seseorang tidak menjaga kesehatan ginjalnya dengan baik yaitu penyakit gagal ginjal. Gagal ginjal adalah penyakit yang mana ginjal tidak dapat melakukan fungsinya dengan semestinya. Gagal ginjal itu sendiri bisa terjadi secara kronis dan akut. Apabila fungsi dari ginjal telah menurun dengan tiba-tiba tetapi bisa kembali secara normal apabila masalah yang terjadi sudah teratasi itu merupakan gagal ginjal akut. Sedangkan gagal ginjal kronis memiliki gejala yang bertahap dan biasanya gagal ginjal kronis tidak menimbulkan gejala di awal yang jelas. Akibatnya fungsi dari ginjal menurun dan penurunannya tidak bisa teramati, secara tiba-tiba saja sudah pada level yang sulit untuk mendapat pertolongan. Penyakit ini merupakan penyakit yang masuk pada penyakit silent killer. Mengapa dikatakan begitu sebab penyakit ini tidak menunjutkan gejala peringatan sebelumnya. Penyakit ini bisa disebabkan dari hal yang sepele seperti dehidrasi, sehingga tubuh mudah terserang infeksi saluran kemih.

Penelitian Global Burden of Disease pada tahun 2010, gagal ginjal kronis adalah penyebab angka kematian yang mendapat peringkat 27 pada tahun 1990 serta menjadi peringkat 18 pada tahun 2010. Pada tahun 2013 Indonesia sendiri mempunyai angka penderita kurang lebih 499.800 penduduk mengalami gagal ginjal serta 1.499.400 penduduk mengalami batu ginjal (Rikesda, 2013). Sedangkan menurut Perhimpunan Nefrologi Indonesia (PERNEFRI) serta Kementrian Kesehatan mengatakn bahwa angka pasien gagal ginjal kronis di Indonesia sebesar 25 sampai dengan 30 juta orang (Candra, 2013). Penyakit gagal ginjal kronis awalnya tidak menunjukkan tanda dan gejala namun dapat berjalan progresif menjadi gagal ginjal. Penyakit ginjal bisa dicegah dan ditanggulangi dan kemungkinan untuk mendapatkan terapi yang efektif akan lebih besar jika diketahui lebih awal. Penyakit gagal ginjal kronis itu sangat sulit dideteksi secara medis sebab para dokter yang ahli di bidangnya ini tidak dapat melakukan pemeriksaan secara visual terhadap pasien. Penyakit gagal ginjal kronis ini dapat di dideteksi dengan melakukan pemeriksaan di laboratorium, salah satunya menggunakan pemeriksaan Glomerular Filtration Rate (GFR).

GFR merupakan tes untuk mengetahui jumlah dari darah yang sudah disaring ginjal pada setiap menit. Walaupun penyakit gagal ginjal kronis dapat dideteksi

menggunakan GFR tetapi proses untuk melakukannya sangat rumit. Terdapat variabel dan rumusan khusus untuk pasien dengan usia tertentu.

Di Indonesia sendiri masyarakat masih belum sadar kalau mereka telah mengalami penyakit ini padahal kalau masih stadium awal masih dapat di terapi tanpa cuci darah. Selain itu dalam melakukan check up membutuhkan waktu yang lama, dikarenakan dalam proses check up ada dua tahapan yaitu proses check lab dan proses konsultasi dokter. Penelitian ini juga berguna dalam membantu tenaga medis dalam mengklasifikasikan secara cepat antara pasien yang normal dan pasien yang terkena penyakit gagal ginjal kronis.

Klasifikasi merupakan suatu cara dalam mengelompokan sebuah dataset ke dalam suatu kelas. Klasifikasi itu sendiri dapat dilakukan dengan berbagai metode seperti Extreme Learning Machine (ELM) serta Learning Vector Quantization (LVQ). Pada penelitian Ersa (2016) menerangkan mengenai klasifikasi keadaan mata berdasarkan sinyal EEG menggunakan Extreme Learning Machine didapat hasil akurasi sebesar 97,95%. Penelitian selanjutnya klasifikasi gagal ginjal kronis menggunakan C4.5 didapat hasil akurasi sebesar 91,50% (Rianto, 2015).

Berdasarkan penjelasan diatas terbukti metode Extreme Learning Machine menghasilkan nilai yang baik dalam hal klasifikasi. Maka dari itu peneliti ingin melakukan penelitian mengenai klasifikasi risiko gagal ginjal kronis menggunakan metode Extreme Learning Macine. Penelitian ini nantinya diharapkan bisa membantu mengklasifikasikan risiko gagal ginjal konis untuk selanjutnya dapat menekan angka prevalensi gagal ginjal kronis di Indonesia. Sehingga kita sebagai masyarakat sadar akan pentingnya kesehatan khususnya terhadap penyakit gagal ginjal kronis. Dengan demikian diharapkan masyarakat mau untuk melakukan pola hidup sehat guna menunjang kesehatan bagi diri mereka masing-masing.

1.2 Rumusan masalah

Mengacu pada penjelasan latar belakang diatas, rumusan masalah yang dapat dikaji berupa:

1. Bagaimana mendapatkan parameter yang optimal dari klasifikasi risiko gagal ginjal kronis menggunakan *Extreme Learning Machine*?
2. Bagaimana mengukur tingkat akurasi yang dihasilkan dari pengklasifikasian risiko gagal ginjal kronis menggunakan *Extreme Learning Machine*?

1.3 Tujuan

Penelitian yang dilakukan oleh penulis memiliki tujuan yaitu:

1. Membangun sebuah perangkat lunak yang dapat mengklasifikasikan risiko gagal ginjal kronis berdasarkan sejumlah inputan yang diberikan oleh user ke perangkat lunak dengan menggunakan metode *Extreme Learning Machine*.
2. Menghitung tingat akurasi hasil klasifikasi risiko gagal ginjal kronis menggunakan metode *Extreme Learning Machine*.

1.4 Manfaat

Manfaat yang ingin diperoleh dari penelitian ini adalah:

1. Mempermudah mendeteksi risiko gagal ginjal kronis berdasarkan pola hidup masyarakat oleh petugas kesehatan.
2. Membantu petugas kesehatan untuk mengklasifikasikan risiko penyakit gagal ginjal kronis.
3. Memudahkan dan mempercepat dalam pengolahan data prevalensi gagal ginjal kronis.
4. Membantu instansi kesehatan dalam rangka *surveillance* penyakit tidak menular khususnya gagal ginjal kronis.

1.5 Batasan masalah

Hal-hal yang menjadi batasan masalah pada penelitian ini yaitu :

1. Penyakit yang dibahas merupakan penyakit gagal ginjal kronis dari data set *Indians Chronic Kidney Disease*.
2. Faktor risiko yang menjadi pengukuran adalah faktor resiko yang berasal dari gaya hidup yang dijalani. Mulai dari age, tekanan darah, specific grafity, albumin, gula, sel darah merah, pus cell, pus cell clumps, bakteri, glukosa darah acak, urea darah, kreatinin serum, natrium, kalium, darah merah, jumlah sel darah putih, jumlah sel darah merah, packed cell volume, tekanan darah tinggi, kencing manis, coronary artery disease, selera, pedal edema, kekurangan darah.
3. Data yang digunakan pada penelitian ini adalah data sekunder rekapitulasi pemeriksaan faktor risiko penyakit tidak menular pada Apollo Hospitals, Managiri, Madurai Main Road, Karaikudi, Tamilnadu, India. Data set diambil dari situs web https://archive.ics.uci.edu/ml/datasets/chronic_kidney_disease.
4. Bahasa pemrograman yang dipakai pada penelitian ini adalah bahasa pemrograman java.

1.6 Sistematika Penulisan

1. BAB I PENDAHULUAN

Menguraikan tentang latar belakang, perumusan masalah, tujuan penelitian ini, manfaat dari penelitian ini, batasan malasah dan sistematika pembahasan penelitan.

2. BAB II KAJIAN PUSTAKA

Menguraikan tentan dasar teori dan referensi yang mendasari proses perancangan dan implementasi perangkat lunak untuk Klasifikasi Risiko Gagal Ginjal Kronis Menggunakan *Extreme Learning Machine*.

3. BAB III METODOLOGI

Terdiri atas metode dan langkah kerja yang dilakukan dalam penelitian implementasi algoritma *Extreme Learning Machine* untuk Klasifikasi Risiko Gagal Ginjal Kronis.

4. BAB IV PERANCANGAN

Membahas perancangan sistem serta proses-proses algoritma *Extreme Learning Machine* untuk klasifikasi Risiko Gagal Ginjal Kronis.

5. BAB V IMPLEMENTASI

Menjelaskan tentang implementasi sistem. Implementasi sistem berisikan Batasan implementasi, proses implementasi, penggunaan algoritma *Extreme Learning Machine*.

6. BAB VI PENGUJIAN DAN ANALISIS

Bab ini berisikan pengujian dari algoritma *Extreme Learning Machine* beserta analisis hasil yang diperoleh.

7. BAB VII PENUTUP

Bab ini berisi tentang kesimpulan dan saran.

BAB 2 TINJAUAN PUSTAKA

Pada bab tinjauan pustaka akan membahas mengenai beberapa teori serta kajian pustaka yang bersangkutan dengan klasifikasi risiko gagal ginjal kronis menggunakan *Extreme Learning Machine.* Dasar teori akan membahas teori mengenai gagal ginjal kronis serta beberapa hal yang berkaitan terhadap penelitian yang sedang diteliti.

2.1 Tinjauan Penelitian

Penelitan tentang diagnosa gagal ginjal kronis maupun risiko gagal ginjal kronis telah banyak bermunculan dengan menggunakan berbagai pendekatan metode. Pada penelitian ini digunakan referensi yang relevan sebagai rujukan dalam proses penelitan nantinya.

Salah satu referensi yang menjadi rujukan pada penelitan ini adalah penelitian yang dilakukan oleh Restu Pranandari (2015) menerangkan tentang faktor risiko penyakit gagal ginjal kronis. Dalam penelitian itu hanya menjelaskan faktor yang berhubungan mengenai penyakit gagal ginjal kronis, antara lain umur, jenis kelamin, riwayat mengenai penyakit hipertensi, riwayat mengenai penyakit diabetes melitus, riwayat penggunaan obat analgetika, anti inflamasi non-steroid, riwayat merokok dan riwayat penggunaan suplemen energi.

Penelitian kedua oleh Muhammad Imanurrahman (2015) pada penelitan ini bertujuan untuk melakukan pengklasifikasian penyakit ginjal menggunakan metode *naive bayes*. Untuk metode penelusuran yang digunakan adalah *forward chaining* dan metode probabilitasnya menggunakan *naive bayes.*

Penelitian ketiga oleh Shahsavari, Bakhsh & Rashidi (2016) pada penelitan ini menggunakan gabungan metode klasifikasi terhadap metode seleksi fitur yang mana didapatkan hasil klasifikasi yang baik serta lebih efisien dari metode klasifikasi yang biasa. Metode yang dapat digunakan yaitu metode *Extreme Learning Machine* untuk klasifikasi serta metode *Particle Swarm Optimization* untuk seleksi fitur. Metode *Extreme Learning Machine* dipilih sebab mempunyai kecepatan perhitungan tinggi serta mempunyai kemampuan generalisasi yang baik. *Extreme Learning Machine* merupakan metode pembelajaran yang mempunyai akurasi yang baik.

Penelitan keempat oleh Siwi, Cholissodin *and* Furqon (2016) mengenai peramalan produksi gula pasir di Pabrik Gula Candi Baru daerah Sidoarjo menggunakan metode *Extreme Learning Machine* dan didapatkan nilai *error* MAPE 3.66%.

Penelitian kelima oleh Ersa (2016) mengenai klasifikasi keadaan mata berdasarkan sinyal EEG menggunakan *Extreme Learning Machine.* Dari penelitian yang telah dilakukan oleh Ersa pada tahun 2016 ini didapatkan nilai akurasi sebesar 97,95%.

Penelitian keenam oleh Rianto (2015) mengenai klasifikasi gagal ginjal kronis menggunakan C4.5 didapatkan hasil akurasi sebesar 91,50%. Tabel 2.1 merupakan ringkasan dari penelitian tersebut.

Table 2.1 Ringkasan penelitian sebelumnya

No.	Judul	Objek	Metode	Hasil
1.	Faktor Risiko Gagal Ginjal Kronis di Unit Hemodialisis RSUD Wates Kulon Progo.	Menggunakan penelitian observasi analitik dengan pendekatan case control dengan penelusuran riwayat pasien apakah ada hubungan antara faktor risiko gagal ginjal kronis dengan kejadian gagal ginjal kronis pada pasien penderita gagal ginjal kronis.	Sampel kasus dalam penelitian adalah pasien yang terdiagnosis mengalami gagal ginjal kronis yang diketahui melalui rekam medik dan wawancara serta pasien tersebut rutin melakukan hemodialisis di RSUD Wates Kulon Progo dan juga Sampel kontrol pasien rawat inap di RSUD Wates Kulon Progo.	Hasil akhirnaya nanti akan diketahui faktor-faktor yang mempengaruhi penyakit gagal ginjal kronis. Akan dijelaskan seberapa besar nilai dari faktor-faktor yang mempengaruhi tersebut.
2.	Pengklasifikasian Penyakit Ginjal Menggunakan Metode *Naive Bayes.*	Obyek yang digunakan pada penelitian ini adalah penyakit ginjal pada manusia.	Metode penelusuran yang digunakan adalah *forward chaining* dan metode probabilitasnya menggunakan *naive bayes.*	*Output* yang dihasilkan berupa hasil diagnosa terhadap penyakit, penyebab penyakit.
3	*Efficient Classification of Parkinson's Disease Using Extreme Learning Machine and Hybrid Particle Swarm Optimization.*	Data pasien penderita penyakit Parkinson.	*Extreme Learning Machine* dan *Hybrid Particle Swarm Optimization.*	Metode ELM dan PSO memiliki akurasi tertinggi yaitu mencapai 88.72% jika dibandingkan dengan

				metode lainnya.
4	Peramalan Produksi Gula Pasir Menggunakan *Extreme Learning Machine* (*ELM*) pada PG Candi Baru Sidoarjo.	Data produksi gula pasir dan jumlah *hidden layer.*	*Extreme Learning Machine.*	Dengan metode *Extreme Learning Machine* didapatkan *error* MAPE yang baik yaitu sebesar 3.66%.
5	Klasifikasi Keadaan Mata Berdasarkan sinyal EEG menggunakan *Extreme Learning Machine.*	Obyek yang digunakan pada penelitian ini adalah data *eyestate*.	*Extreme Learning Machine.*	Dengan metode *Extreme Learning Machine* dihasilkan nilai akurasi sebesar 97,95%.
6	Rancang Bangun Aplikasi Pendeteksi Penyakit Ginjal Kronis dengan Menggunakan Algoritme C4.5	Obyek yang digunakan pada penelitian ini adalah penyakit gagal ginjal kronis.	Algoritme C4.5.	Dengan menggunakan Algoritme C4.5 didapat hasil akurasi sebesar 91,50%.

Sumber : (Restu Pranandari, 2015), (Muhammad Imanurrahman, 2015), (Rianto, 2015), (Shahsavari, et al., 2016), (Siwi, Cholissodin *and* Furqon 2016), (Ersa, 2016).

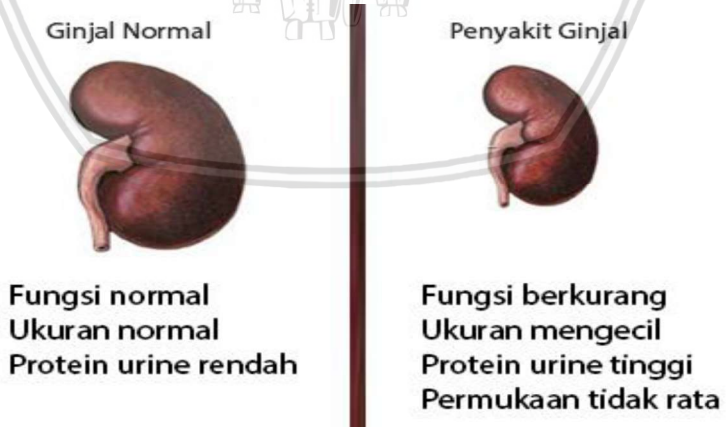
Tabel 2.1 merupakan tabel yang berisi penelitian-penelitian sebelumnya terkait penyakit gagal ginjal kronis yang diteliti menggunakan berbagai metode yang relevan dengan penyakit gagal ginjal kronis, dan terkait tentang metode *Extreme Learning Machine* dimana metode ini dipilih karena mekanisme proses *learning speed* yang digunakan sangat cepat dan efisien. Sehingga apabila dalam melakukan proses perhitungan membutuhkan waktu yang lama, dengan menggunakan metode ini hasilnya akan lebih efisien. Oleh sebab itu pada penelitian ini akan dicoba untuk mengklasifikasikan risiko gagal ginjal kronis dengan menggunakan merode *Extreme Learning Machine.*

2.2 Klasifikasi

Klasifikasi adalah sebuah aspek pada data mining yang mana banyak digunakan untuk berbagai macam masalah pada sebuah penelitian. Klasifikasi adalah cara untuk melakukan indentifikasi objek ke suatu kelas ataupun kategori tertentu. Dalam hal ini klasifikasi memiliki tujuan dalam membagi sebuah objek masuk ke salah satu kategori yang biasa disebut dengan kelas. Klasifikasi sering juga dipergunakan dalam melakukan prediksi. Klasifikasi biasa bekerja melalui cara penentuan model dari data *training* yang mana kelas labelnya sudah diketahui.

2.3 Gagal Ginjal Kronis

Ginjal adalah dua buah organ yang berbentuk kacang yang berada di sebelah kiri serta kanan tubuh di *vertebrata.* Ginjal berda di bagian belakang rongga perut pada ruang *retroperitoneal*. Pada orang dewasa panjangnya sekitar 11 cm. Ginjal menerima darah dari pembuluh ateri ginjal. Ginjal tersusun dari *nefron*. *Nefron* merupakan unit struktural dan fungsional dari ginjal. Setiap ginjal mengandung sekitar satu juta *nefron*. *Nefron* menggunakan empat proses untuk mengubah plasma darah yang mengalir ke ginjal. Yang meliputi filtrasi, reabsorpsi, sekresi, dan ekskresi. Dengan melalui proses ini ginjal berpartisipasi dalam pengendalian volume berbagai kompartemen cairan tubuh, osmolalitas cairan, keseimbangan asam basa, berbagai konsentrasi elektrolit, dan pengangkatan toksin. Gagal ginjal kronis adalah penyakit yang mana ginjal mengalami penurunan fungsi. Sampai saat ini, gagal ginjal kronis masih menjadi perhatian utama pemerintah karena meningkatnya prevalensi gagal ginjal kronis dan banyaknya pasien gagal ginjal kronis yang belum mendapatkan pertolongan medis Gambar 2.1 merupakan ginjal normal dan gagal ginjal kronis.



Gambar 2.1 Ginjal Normal dan Gagal Ginjal Kronis

2.3.1 Faktor Risiko Gagal Ginjal Kronis

Faktor risiko adalah sebuah kebiasaan atau atribut individu yang umum dialami seorang penderita. Atribut individu yang dimaksud bisa umur, jenis kelamin serta riwayat sebuah penyakit yang pernah di derita. Kebiasaan yang biasa

menjadi risiko bisa berupa merokok, kebiasaan menggunakan obat analgetika, penggunaan minuman suplemen energi. Pada penyakit gagal ginjal kronis terdapat beberapa risiko, diantaranya (Restu Pranandari, 2015) :

1. Kebiasaan merokok. Perokok aktif memiliki probabilitas yang lebih tinggi untuk terkena penyakit gagal ginjal kronis daripada orang yang bukan perokok.
2. Kebiasaan menggunakan obat analgetika. Pengguna obat analgetika aktif memiliki probabilitas yang lebih tinggi untuk terkena penyakit gagal ginjal kronis daripada orang yang jarang menggunakan.
3. Kebiasaan mengkonsumsi minuman suplemen. Orang yang melakukan kebiasaan ini memiliki probabilitas yang lebih tinggi untuk terkena penyakit gagal ginjal kronis daripada orang yang jarang melakukannya.
4. Riwayat penyakit diabetes militus dan hipertensi. Penyakit ini bisa menyebabkan munculnya penyakit gagal ginjal kronis.

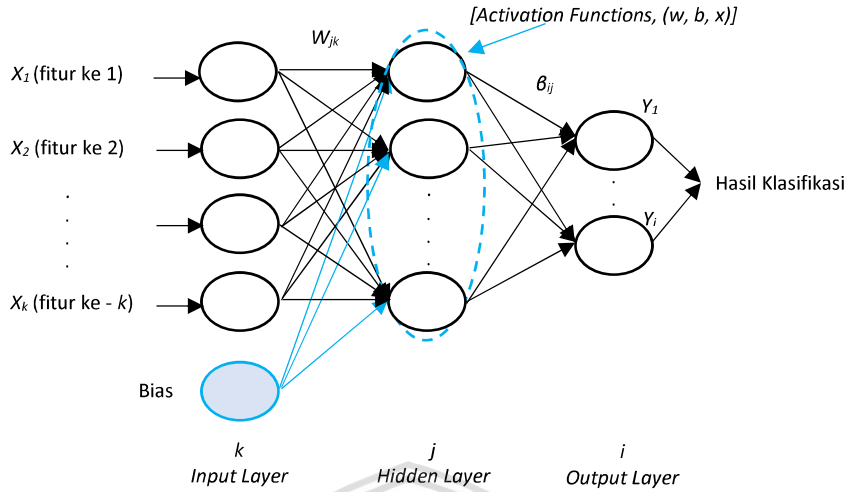
Umur dan jenis kelamin. Laki-laki dengan umur kurang dari 60 tahun dan perempuan umur lebih dari 60 tahun memiliki kemungkinan yang besar untuk terkena penyakit gagal ginjal kronis.

2.4 *Extreme Learning Machine*

Extreme Learning Machine (ELM) adalah sebuah algoritme pembelajaran pada jaringan saraf tiruan yang berjenis *single hidden layer feedforward network* (SLFN non-parametrik). Algoritme ini diperkenalkan tahun 2004 oleh Huang. Algoritme ini bisa melakukan proses generalisasi dengan waktu yang cepat di banding *Support Vector Machine* (SVM) dan *Backpropagation* (Huang, et al., 2006).

Menurut Bartlett, *feedforward neural network* bisa melakakuan penjangkauan kesalahan pelatihan begitu kecil. Nilai bobot semakin kecil maka proses generalisasi pada jaringan menjadi lebih baik (Huang, et al., 2012). Algoritme *Extreme Learning Machine* diciptakan tujuannya untuk menyempurnakan kelemahan jaringan saraf tiruan *feedforward* dalam hal *learning speed*. *Extreme Learning Machine* adalah jaringan saraf tiruan yang bisa dilatih untuk melakukan sebuah fungsi kompleks pada berbagai macam bidang, yang paling utama yaitu untuk klasifikasi mengenai hal-hal yang begitu kompleks.

Parameter yang digunakan untuk algoritme *Extreme Learning Machine* seperti halnya nilai *input weight* dan *input* bias dipilih dengan cara *random*, sehingga dengan demikian waktu yang dibutuhkan algoritme *Extreme Learning Machine* dalam pembelajaran sangat kecil dan cepat. Selain waktu pembelajaran yang cepat nilai akurasi yang dihasilkan juga tinggi. Gambar 2.2 merupakan struktur metode ELM.



Gambar 2.2 Struktur ELM

Sumber: (Yaseen, et al. 2017)

Pada struktur *Extreme Learning Machine* pada gambar 2.2 dijelaskan bahwa suatu proses pada algoritme ini adalah nilai data masukan akan diproses bersama nilai *input weight* serta nilai bias, setelah itu diproses pada setiap nuron. Hasil dari sebuah proses *Extreme Learning Machine* berupa *output layer.* Langkah perhitungan algoritme *Extreme Learning Machine* dalam penelitian ini adalah yang pertama melakukan proses *normalisasi* data, setelah itu data hasil proses *normalisasi* akan dilakukan proses *training* serta proses *testingi* pada algoritme *Extreme Learning Machine.*

2.4.1 Normalisasi Data

Normalisasi data ini dilakukan karena *range* nilai *input* tidak sama. Pada penelitian yang sedang dilakukan ini nilai *range* data yang digunakan akan disesuaikan menggunakan *normalisasi* data. Persamaan 2.1 merupakan rumus normalisasi data dengan metode *MinMax Normalization* (Jain & Bhandare, 2011).

$$d' = \frac{d - min}{\max - min} \quad (2.1)$$

Keterangan:

d' = *Normalisasi* data.

d = Nilai asli.

min = Nilai minimal data.

max = Nilai maksimal data.

2.4.2 Proses *Training*

Proses yang pertama dilakukan untuk algoritme *Extreme Learning Machine* yaitu proses *training*. Pada proses *training* akan membutuhkan *input* data. *Input* data ini akan digunakan untuk data *training set* serta *output* target untuk proses

klasifikasi (Imam Cholissodin, 2017). Langkah tahapan proses *training* algoritme *Extreme Learning Machine* adalah:

1. Pemilihan secara *random input weight* (*W*) dengan nilai *range* -1 sampai dengan 1 serta nilai *input* bias (*b*) dengan nilai *range* 0 sampai dengan 1.
2. Hitung nilai matriks keluaran *neuron hidden layer* (*Hinit*). Persamaan 2.2 berikut ini merupakan persamaan untuk menghitung keluaran *hidden neuron.*

$$\boldsymbol{H_{init} = (X_{train} * W^T) + b} \quad (2.2)$$

Keterangan:

H_{init} = Keluaran *neuron hidden layer*.

X_{train} = Nilai masukan data data *training.*

w = Nilai bobot masukan.

b = Nilai dari bias.

3. Hitung nilai matriks keluaran *neuron hidden layer* dengan menggunakan fungsi aktivasi sigmoid biner *(H).* Persamaan 2.3 berikut ini merupakan persamaan untuk menghitung *output hidden neuron* dengan fungsi aktivasi sigmoid biner.

$$\boldsymbol{H(x) = \frac{1}{1+e^{-x}}} \quad (2.3)$$

4. Hitung nilai dari *Moore-Penrose* (H^+). Persamaan 2.4 berikut ini merupakan persamaan untuk menghitung matriks *Moore-Penrose.*

$$\boldsymbol{H^+ = (H^T . H)^{-1} * H^T} \quad (2.4)$$

5. Hitung nilai *output weight (β).* Jika kelas sebanyak dua, maka panjang kolom target sebanyak dua. Dimana kolom target dituliskan [1,-1] jika termasuk kelas satu dan kolom target dituliskan [-1,1] jika termasuk kelas dua. Persamaan 2.5 berikut ini merupakan persamaan untuk menghitung *output weight*.

$$\boldsymbol{\beta = H^+ . T} \quad (2.5)$$

6. Hitung nilai dari *output layer (Y).* Persamaan 2.6 berikut ini merupakan persamaan untuk menghitung *output layer.*

$$\boldsymbol{Y = H . \beta} \quad (2.6)$$

2.4.3 Proses *Testing*

Setelah proses *training* sudah selesai dan di dapat nilai *output layer*, selanjutnya akan dilanjutkan ke proses *testing*. Langkah tahapan proses *testing* dengan algoritme *Extreme Learning Machine* adalah:

1. Gunakan nilai *input weight (W)* yang nilainya sama dengan proses *training.* Gunakan nilai bias *(b)* yang didapatkan secara *random* dengan *range* 0 sampai dengan 1.

2. Hitung nilai matriks keluaran *neuron hidden layer (Hinit).* Persamaan 2.7 berikut ini merupakan persamaan untuk menghitung keluaran *neuron hidden layer.*

Pada proses ini bertujuan untuk mendapatkan hasil klasifikasi dan mengevaluasi medote ELM dari hasil proses *training* sebelumnya. Proses *testing* dilakukan menggunakan *input weight,* bias dan *output weight* yang didapatkan dari proses *training.* Berikut adalah langkah-langkah proses *testing* (Huang, et al., 2006):

1. Langkah pertama menggunakan bobot *input* (Wi) yang sama seperti proses *training* dan *hidden bias* (bi) dengan *range* 0 sampai 1.
2. Menghitung matriks *output hidden neuron* (Hinit). Persamaan 2.7 berikut ini merupakan persamaan untuk menghitung keluaran *hidden neuron.*

$$\boldsymbol{H_{init}} = (\boldsymbol{X_{test}} * \boldsymbol{W^T}) + \boldsymbol{b} \quad (2.7)$$

Keterangan:

H_{init} = Keluaran *neuron hidden layer.*

X_{test} = Nilai masukan *testing.*

w = Nilai bobot masukan.

b = Nilai dari bias.

3. Hitung nilai matriks keluaran *neuron hidden layer* dengan menggunakan fungsi aktivasi sigmoid biner *(H).* Persamaan 2.8 berikut ini merupakan persamaan untuk menghitung keluaran *hidden neuron* dengan fungsi aktivasi sigmoid biner.

$$\boldsymbol{H(x)} = \frac{\mathbf{1}}{\mathbf{1}+\boldsymbol{e^{-x}}} \quad (2.8)$$

4. Hitung nilai *output layer (Y)* proses *testing.* Persamaan 2.9 berikut ini merupakan persamaan untuk menghitung keluaran *output layer.*

$$\boldsymbol{Y} = \boldsymbol{H}.\boldsymbol{\beta} \quad (2.9)$$

Keterangan:

β = Bobot keluaran yang didapat pada proses *training.*

2.4.4 Akurasi Sistem

Akurasi adalah sebuah nilai yang mana nilai ini nantinya akan digunakan dalam proses analisis untuk tingkat keberhasilan dari sebuah sistem dalam menangani masalah. Nilai akurasi biasa diambil dengan cara melakukan perbandingan dari jumlah data yang benar dengan jumlah nilai data keseluruhan yang ada pada sistem. Persamaan 2.10 berikut ini merupakan persamaan untuk menghitung nilai akurasi pada hasil klasifikasi (Simanjuntak, et al., 2017):

$$\textbf{Akurasi} = \frac{\textbf{DT}}{N} * \textbf{100}\% \quad (2.10)$$

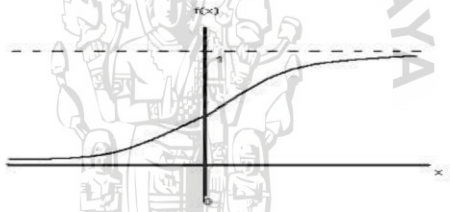
Keterangan:
DT = Jumlah benar pada sistem.
N = Jumlah semua data pada sistem.

2.5 Fungsi Aktivasi

Fungsi aktivasi adalah salah satu fungsi yang memiliki kemampuan untuk memproses sebuah masukan kemudian melakukan transpormasi pada masukan tadi supaya menjadi sebuah keluaran tertentu. Pada jaringan saraf tiruan sebuah informasi akan dijadikan menjadi sebuah masukan. Kemudian nilai dari masukan akan diproses melalui suatu fungsi perambatan. Dalam fungsi perambatan nilai dari sejumlah masukan akan dilakukan penjumlahan, hasilnya akan dilakukan proses perbandingan dengan nilai ambang tertentu sesuai fungsi aktivasi yang ada di setiap neuron. Fungsi aktivasi yang digunakan pada algoritme Extreme Learning Machine (Srimuang & Intarasothonchun, 2015):

1. Fungsi *sigmoid biner*

   Fungsi sigmoid biner mempunyai nilai keluaran dengan *range* 0 sampai dengan 1. Persamaan 2.11 berikut ini merupakan persamaan untuk menghitung sigmoid biner. Gambar 2.3 merupakan fungsi *sigmoid biner.*



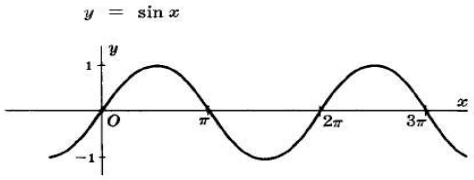
Gambar 2.3 Fungsi *Sigmoid Biner*

$$\boldsymbol{y = f(x) = \frac{1}{1+e^{-x}}} \quad (2.11)$$

   Keterangan:
   F(x) = *Sigmoid* biner.
   e^{-x} = *Exponensial*.

2. Fungsi *sin*

   Nilai rentang pada fungsi *sin* dengan *range* -1 sampai dengan 1. Persamaan 2.12 berikut ini merupakan persamaan untuk menghitung fungsi *sin.* Gambar 2.4 merupakan fungsi *sin.*



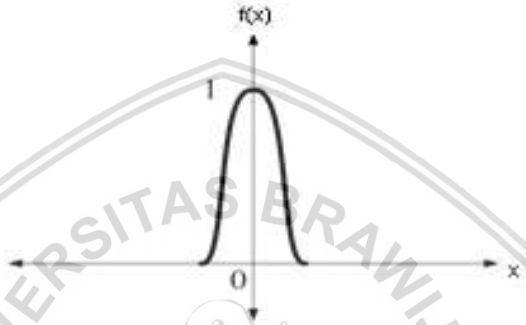
Gambar 2.4 Fungsi *Sin*

$$f(x) = \sin x \tag{2.12}$$

Keterangan:
f(x) = Ativasi *sin.*
sin x = nilai *sin* pada data *x*.

3. Fungsi *radial basis*

Fungsi *radial basis* adalah salah satu fungsi aktivasi yang digunakan dalam jaringan syaraf tiruan. Persamaan 2.13 berikut ini merupakan persamaan untuk menghitung fungsi *radial basis*. Gambar 2.5 merupakan fungsi *radial basis.*



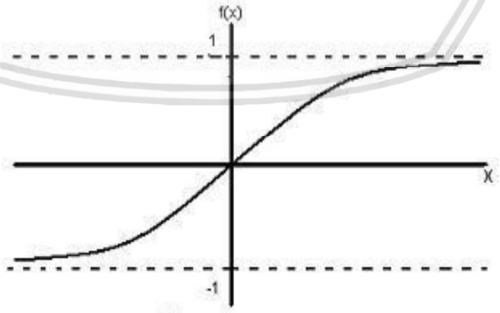
Gambar 2.5 Fungsi *Radial Basis*

$$H = \exp(-(Hinit)^2) \tag{2.13}$$

Keterangan:
H = Fungsi aktivasi *radial basis*
Hinit = Nilai *hidden layer*

4. Fungsi *bipolar*

Nilai rentang pada fungsi bipolar dengan *range* -1 sampai dengan 1. Persamaan 2.14 berikut ini merupakan persamaan untuk menghitung fungsi bipolar. Gambar 2.6 merupakan fungsi *bipolar.*



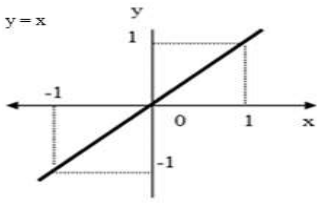
Gambar 2.6 Fungsi *Bipolar*

$$H = \frac{1-exp^{-Hinit}}{1+exp^{-Hinit}} \tag{2.14}$$

Keterangan:
H = Fungsi aktivasi *bipolar*
Hinit = Nilai *hidden layer*

5. Fungsi linier

   Fungsi aktivasi linier merupakan salah satu fungsi aktivasi yang digunakan dalam jaringan syaraf tiruan. Persamaan 2.15 merupakan rumus dari aktivasi linier. Gambar 2.7 merupakan fungsi aktivasi linier.



Gambar 2.7 Fungsi Aktivasi Linier

$$\boldsymbol{H = Hinit} \tag{2.15}$$

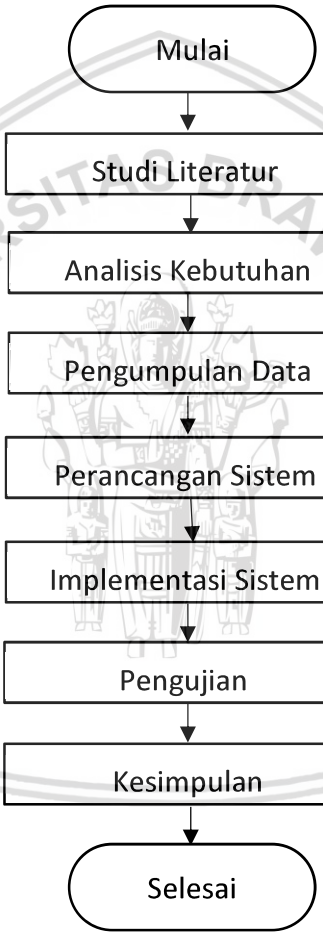
Keterangan:

H = Fungsi aktivasi linier

Hinit = Nilai *hidden layer*

BAB 3 METODOLOGI

Pada penelitian yang dilakukan ini termasuk penelitian *non-implementatif* analitik. Penelitian ini merupakan suatu penelitian yang mengutamakan dalam penggalian sebuah informasi yang sudah ada serta bertujuan dalam pengidentifikasian elemen penting pada sebuah objek penelitian yang mana sebagai dasar dalam mengambil sebuah keputusan selanjutnya. Pada bab metodologi membahas mengenai langkah - langkah penggunaan untuk pembuatan sebuah aplikasi klasifikasi risiko gagal ginjal kronis menggunakan *Extreme Learning Machine*. Pada gambar 3.1 berikut ini adalah struktur metodologi yang akan digunakan dalam penelitian.



Gambar 3.1 Diagram Blok Metodologi Penelitian

3.1 Studi Literatur

Pembelajaran terhadap literatur dari macam-macam bidang ilmu yang berhubungan terhadap penyakit Gagal Ginjal Kronis menggunakan metode *Extreme Learning Machine*, diantaranya :

1. Penyakit Gagal Ginjal Kronis
2. Metode *Extreme Learning Machine*
3. Kalsifikasi mengunakan E*xtreme Learning Machine*

4. Proses pengujian

Literatur tersebut didapatkan dalam buku, jurnal *e-book* serta penelitan sebelumnya.

3.2 Analisis Kebutuhan

Analisis kebutuhan sistem mempunyai tujuan untuk memperoleh semua kebutuhan yang akan diperlukan oleh perangkat lunak yang akan di bangun. Analisa kebutuhan dilakukan dengan mengidentifikasi kebutuhan sistem dan apa saja yang terlibat didalamnya. Perangkat lunak yang akan dibuat pada penelitian ini berfokus pada pengujian algoritma Extreme Learning Machine. Analisa kebutuhan perangkat lunak terdiri dari kebutuhan antar muka dan kebutuhan fungsional.

3.2.1 Kebutuhan Antarmuka

Kebutuhan antarmuka dalam sistem yang dibangun antara lain:

1. Perangkat lunak yang dibangun harus memiliki tampilan yang *familiar* untuk *user*.
2. Perangkat lunak yang dibangaun harus mampu menampilkan ringkasan data yang telah diproses sebelumnya oleh sistem.

3.2.2 Kebutuhan Fungsional

Fungsi dalam sistem yang dibangun antara lain:

1. Perangkat lunak harus mampu membaca data latih serta data uji.
2. Perangkat lunak harus mampu melakukan proses pelatihan dengan menggunakan algoritma *Extreme Learning Machine*.
3. Perangkat lunak harus mampu melakukan klasifikasi menggunakan metode *Extreme Learning Machine*.

3.3 Objek Penelitian

Penelitian ini menggunakan data dari data set *Indians Chronic Kidney Disease*. Yang mana jumlah data dalam data set *Indians Chronic Kidney Disease* ini berjumlah 400 data. Setiap data dalam data set *Indians Chronic Kidney Disease* terdiri dari 25 fitur termasuk kelasnya. Kelas dalam data set *Indians Chronic Kidney Disease* terdiri dari 2 kelas yaitu kelas gagal ginjal kronis dan kelas tidak gagal ginjal kronis.

3.4 Pengumpulan Data

Penyakit yang dibahas merupakan penyakit gagal ginjal kronis dari *dataset Indians Chronic Kidney Disease*. Data yang diolah pada penelitian ini merupakan data sekunder rekapitulasi hasil pemeriksaan faktor penyakit tidak menular pada *Apollo Hospitals*, Managiri, Madurai Main Road, Karaikudi, tamilnadu, India. *Dataset* diperoleh melalui situs web https://archive.ics.uci.edu/ml/datasets/chronic_kidney_disease.

Nantinya datanya yang digunakan terdiri dari 155 data. Data akan dilakukan pembagian yang mana sebagian akan dijadikan sebagai data *training* dan sebagian akan di jadikan sebagai data *testing.* Data ini terdiri dari 25 fitur termasuk fitur kelas. O*utput* dari data risiko gagal ginjal kronis adalah pasien yang menderita gagal ginjal kronis serta pasien tidak menderita gagal ginjal kronis. Tabel 3.1 merupakan informasi *dataset Indians Chronic Kidney Disease.*

Tabel 3.1 Informasi data set *Indians Chronic Kidney Disease*

No	Nama Atribut	Data dari atribut
1	*Age*	2-90
2	*Blood Pressure*	*Blood Pressure* in mm/Hg
3	*Specific Gravity*	1.005, 1.010, 1.015, 1.020, 1.025
4	*Albumin*	0, 1, 2, 3, 4, 5
5	*Sugar*	0, 1, 2, 3, 4, 5
6	*Red Blood Cells*	*Normal, abnormal*
7	*Pus Cell*	*Normal, abnormal*
8	*Pus Cell Clumps*	*Present, notpresent*
9	*Bacteria*	*Present, notpresent*
10	*Blood Glucose Random*	*Blood Glucose Random in* mgs/dl
11	*Blood* Urea	*Blodd* Urea *in* mgs/dl
12	Serum *Creatinine*	Serum *Creatinine* in mgs/dl
13	Sodium	Sodium *in* mEq/L
14	Potassium	Potassium *in* mEq/L
15	Hemoglobin	Hemoglobin *in* gms
16	*Packed Cell Volume*	*Numerical*
17	*White Blood Cell*	*White Blood Cell in cells/cumm*
18	*Red Blood Cell*	*Red Blood Cell in millions/cmm*
19	*Hypertension*	*Yes, no*
20	Diabetes melitus	*Yes, no*
21	*Coronary Artery Disease*	*Yes, no*
22	*Appetite*	*Good, poor*
23	*Pedal Edema*	*Yes, no*
24	Anemia	*Yes, no*
25	*Class*	Ckd, notckd

Dalam *dataset* ini terdapat data yang mengalami *missing value* dari setiap atribut yang ada. Tabel 3.2 merupakan informasi dari data *missing value*.

Tabel 3.2 Data *Missing Value*

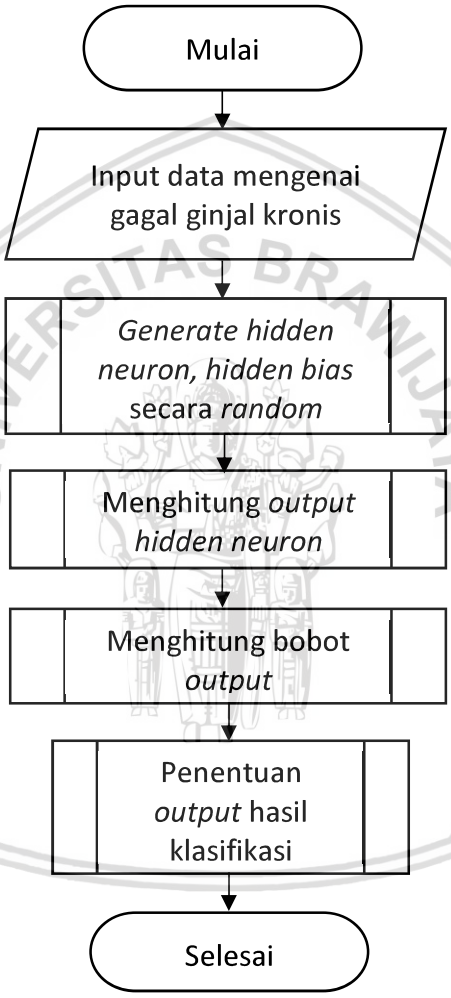
Atribut	**Jumlah *missing value***
1	9
2	12
3	47
4	46
5	49
6	152
7	65
8	4
9	4
10	44
11	19
12	17
13	87
14	88
15	52
16	70
17	105
18	130
19	5
20	2
21	2
22	1
23	1
24	1
25	0

Karena adanya data yang *missing value* maka untuk penelitian ini akan digunakan data yang tidak terdapat *missing value*. Dari data set yang berjumlah 400 data terdapat 155 data yang tidak mengalami *mising value*. Dalam penelitian

ini tidak menggunakan data yang mengalami *missing value* karena *missing value* diisi dengan tanda tanya serta tidak terdapat keterangan lebih lanjut.

3.5 Perancangan Sistem

Dalam perancangan sistem akan dibuat sebuah rancangan langkah kerja pada sistem secara keseluruhan. Secara umum perancangan pada sistem antara lain deskripsi untuk sistem, proses *Extreme Learning Machine*, perhitungan manualisasi, perancangan antarmuka dan pengujian akurasi sistem. Gambar 3.2 merupakan diagram perancangan sistem.



Gambar 3.2 Diagram Perancangan Sistem

3.6 Implementasi Sistem

Implementasi pada sistem sesuai dengan perancangan yang sudah dibuat yang meliputi:

1. Implementasi pada *interface.*
2. Implementasi pada *Microsoft Excel* bertujuan dalam mempermudah proses *input* data.

3. Implementasi pada algoritma dengan memproses perhitungan metode *Etreme Learning Machine* ke dalam bahasa pemrograman *Java* serta menggunakan *software Neatbeans.*
4. Implementasi sistem ini akan menghasilkan *output* mengenai dua macam kriteria klasifikasi yaitu termasuk gagal ginjal kronis dan tidak termasuk gagal ginjal kronis.

3.7 Pengujian

Pengujian sistem dilakukan bertujuan dalam mengetahui sistem apakah berjalan lancar sesuai kebutuhan yang sudah ditentukan beserta mendapatkan hasil dari akurasi mengenai sistem.

3.8 Kesimpulan

Penarikan sebuah kesimpulan diambil setelah semua tahapan sudah dilakukan yaitu tahapan perancangan, tahapan implementasi serta tahapan pengujian sistem. Kesimpulan didapat berdasarkan dari hasil proses pengujian sistem serta analisis mengenai sistem yang mana mempunyai tujuan untuk menjawab dari rumusan masalah yang sudah ditentukan sebelumnya. Selain dari kesimpulan terdapat sebuah tahap akhir yaitu penulisan sebuah saran supaya dapat dipergunakan oleh pembaca atau peneliti selanjutnya untuk acuan serta memperbaiki kekurangan dari penelitian dan juga untuk pengembangan penelitian selanjutnya.

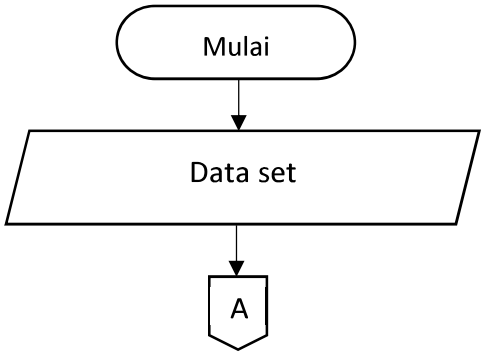
BAB 4 PERANCANGAN

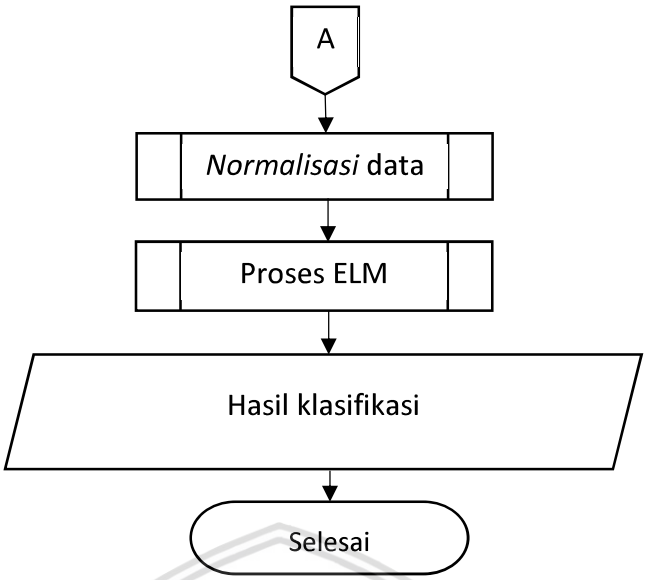
4.1 Formulasi Permasalahan

Dalam kasus ini untuk permasalahan yang akan diselesaikan adalah tentang klasifikasi gagal ginjal kronis menggunakan metode *Extreme Learning Machine.* Yang mana *input* dari klasifikasi ini yaitu *dataset Indians Chronic Kidney Disease.* Data yang dipergunakan untuk penelitian ini merupakan data sekunder rekapitulasi pemeriksaan faktor risiko penyakit tidak menular pada Apollo Hospitals, Managiri, Madurai Main Road, Karaikudi, Tamilnadu, India. *Dataset* yang digunakan ini merupakan data yang diperoleh melalui situs web https://archive.ics.uci.edu/ml/datasets/chronic_kidney_disease. Untuk parameter perhitungan yang akan digunakan yaitu jumlah fitur, fungsi aktivasi serta jumlah *neuron hidden layer.* Dalam penelitian yang dilakukan ini *dataset* akan dibagi menjadi dua bagian yaitu data *training* serta data *testing*. Sebelumnya *dataset* akan dilakukan normalisasi menggunakan *MinMax Normalization.* Setelah itu melakukan perhitungan dengan menggunakan metode *Extreme Learning Machine,* kemudian menentukan nilai akurasi dari hasil yang didapatkan. Untuk *output* yang akan diperoleh berupa klasifikasi dari gagal ginjal kronis .

4.2 Penyelesaian Permasalahan Klasifikasi Risiko Gagal Ginjal Kronis

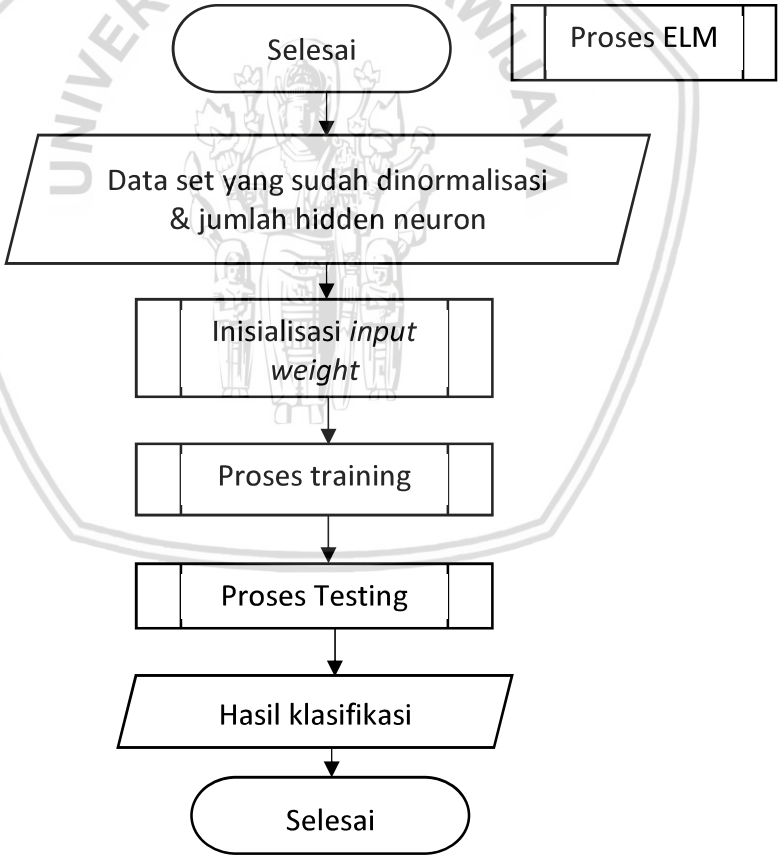
Tujuan dari klasifikasi risiko gagal ginjal kronis yaitu untuk bisa mendapatkan hasil klasifikasi yang tetap sesuai kelas yang sudah tersedia dengan nilai akurasi yang besar. Dalam proses manualisasi klasifikasi menggunakan metode *Extreme Learning Machine* ini menggunakan data input dari *Indians Chronic Kidney Disease.* Dimana data yang digunakan sebanyak 25 data. Dari ke 25 data tadi di bagi menjadi dua bagian 20 data sebagai data *training* dan 5 data sebagai data *testing*. Untuk diagram alur dalam proses klasifikasi gagal ginkal kronis menggunakan ELM ditunjukkan pada gambar 4.1. Dimana diagram alur ini menjelaskan alur dari metode ELM yang nantinya akan digunakan untuk proses *training* dan proses *testing* hingga menghasilkan hasil klasifikasi. Dalam hal ini langkah pertama data akan dinormalisasi terlebih dahulu sebab data dari *Indians Chronic Kidney Disease* mengandung data yang mempunyai *range* yang berbeda-beda. Setelah didapat hasil dari klasifikasi, proses terakhir yaitu mencocokkan hasil dari klasifikasi dengan data aslinya. Gambar 4.1 merupakan diagram alur dalam keseluruhan proses klasifikasi gagal ginjla kronis.





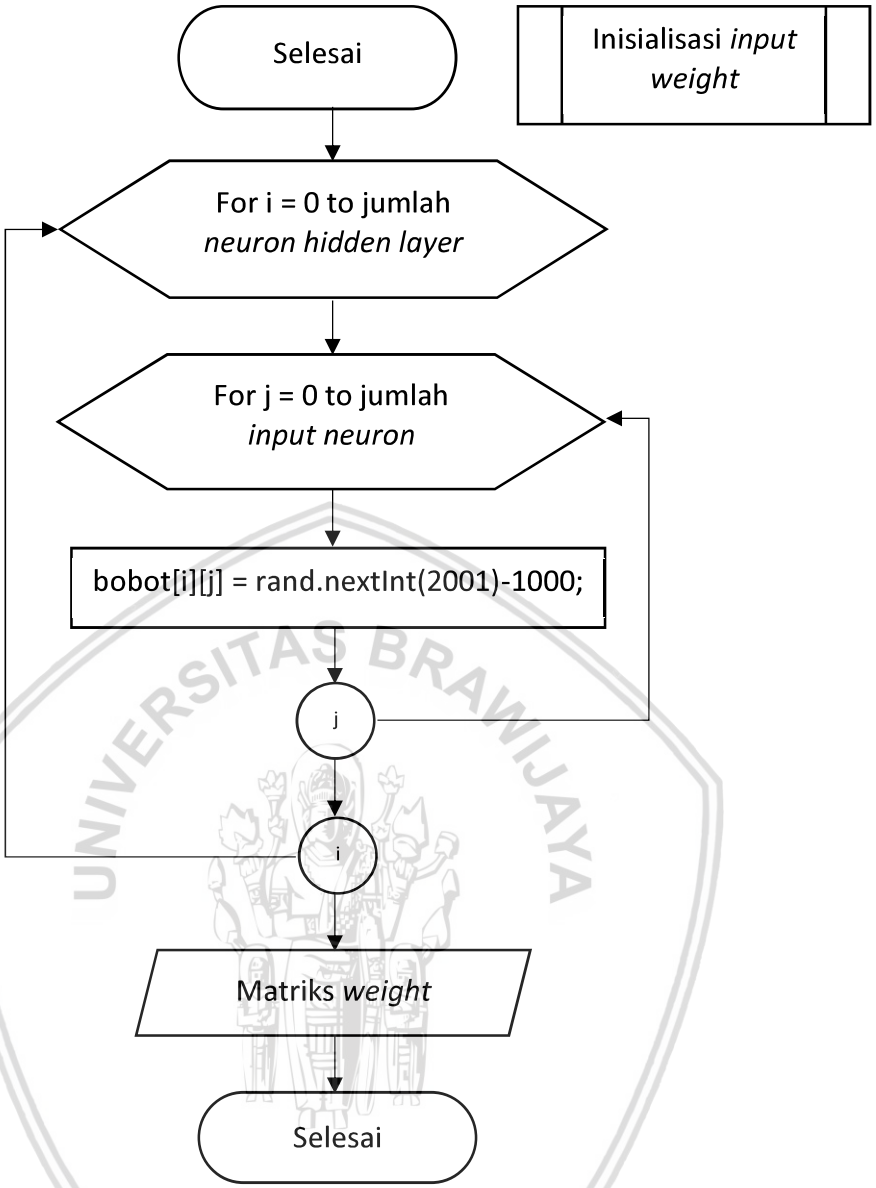
Gambar 4.1 Diagram Alur Sistem

Selanjutnya untuk mengetahui alur dari proses ELM dapat dilihat pada Gambar 4.2 tentang alur proses ELM. Gambar 4.2 merupakan alur proses ELM.



Gambar 4.2 Alur Proses ELM

Kemudian masuk ke proses inisialisasi *input weight*, untuk mengetahui alur dari proses inisialisasi *input weight* dapat dilihat pada Gambar 4.3 tentang alur inisialisasi *input weight*. Gambar 4.3 merupakan alur inisialisasi *input weight.*



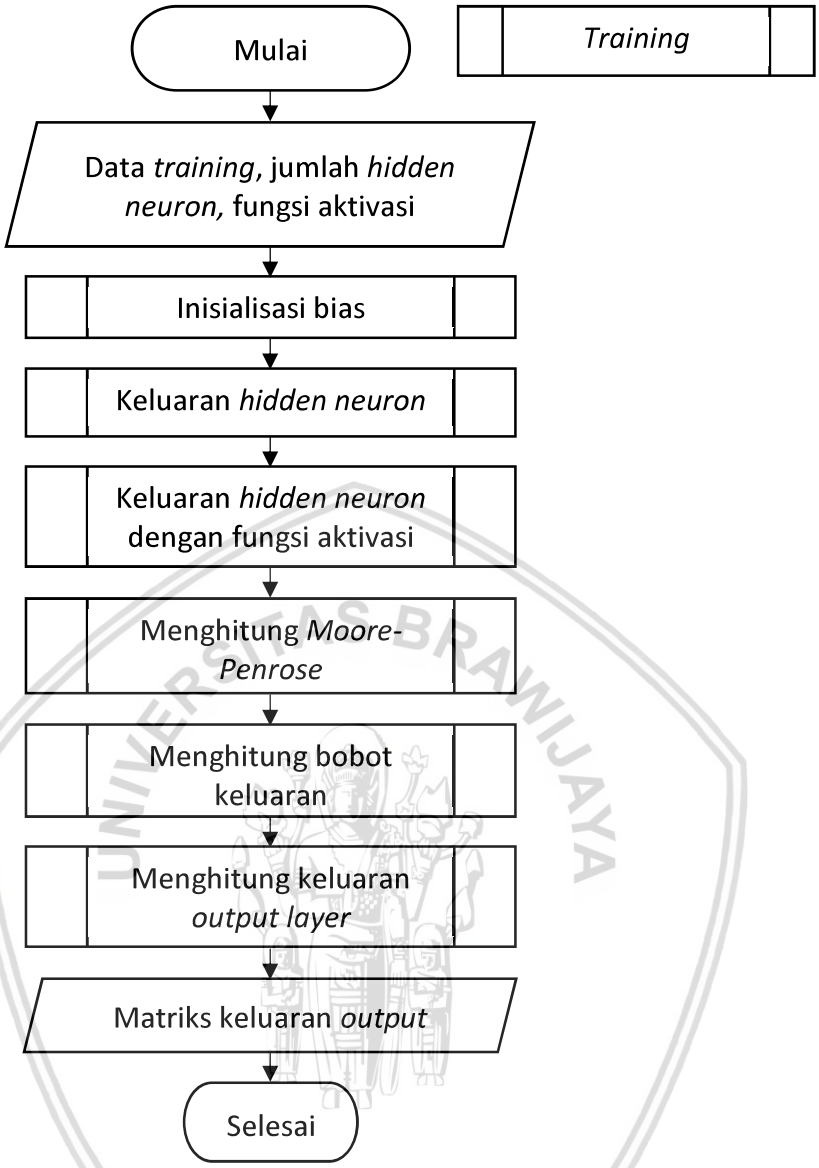
Gambar 4.3 Alur Proses Inisialisasi *Input Weight*

Langkah proses yang terjadi pada inisialisasi *input weight* berdasar pada Gambar 4.3 yaitu:

1. Proses perulangan j sebanyak dari nilai jumlah *input neuron.*
2. Proses perulangan i sebanyak dari nilai jumlah *neuron hidden layer.*
3. Proses *input weight* dengan cara *random* dengan nilai *range* -1 sampai 1.

4.2.2 Proses *Training*

Proses yang pertama dilakukan untuk algoritme Extreme Learning Machine yaitu proses training. Pada proses training akan membutuhkan input data. Input data ini akan digunakan untuk data training set serta output target untuk proses klasifikasi (Imam Cholissodin, 2017). Pada Gambar 4.4 merupakan diagram alur proses *training.*



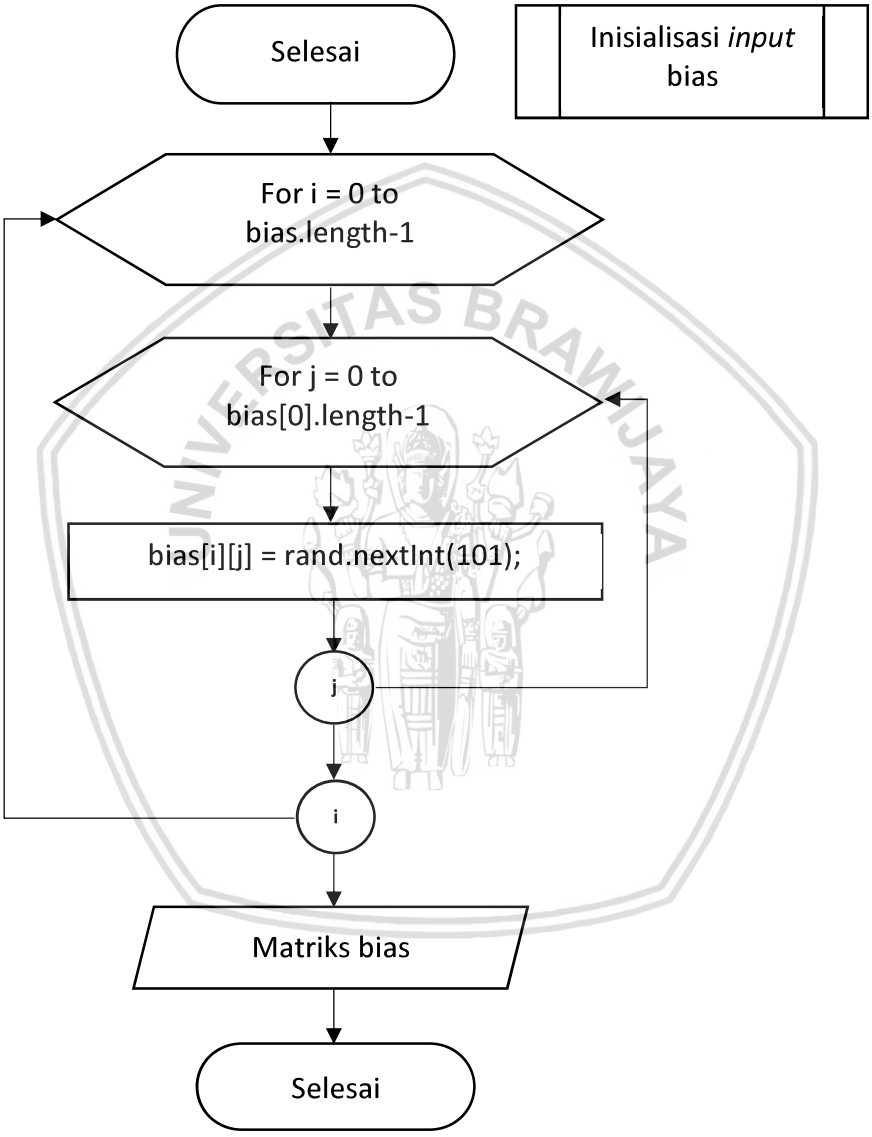
Gambar 4.4 Diagram Alur Proses *Training*

Langkah tahapan proses training algoritme Extreme Learning Machine adalah:

1. Sistem akan menerima masukan data *training,* nilai *neuron hidden layer* serta fungsi aktivasi.
2. Inisialisasi nilai bias menggunakan *range* nilai 0 sampai dengan 1. Proses alur diagram inisialisasi bias bisa dilihat di Gambar 4.5.
3. Proses perhitungan keluaran *neuron hidden layer.* Proses alur diagram keluaran *neuron hidden layer* bisa dilihat di Gambar 4.7.
4. Proses perhitungan keluaran *neuron hidden layer* dengan fungsi aktivasi. Proses alur diagram *neuron hidden layer* dengan fungsi aktivasi bisa dilihat di Gambar 4.8.

5. Proses perhitungan *moore-penrose*. Proses alur diagram *moore-penrose* bisa diliahat di Gambar 4.10.
6. Proses perhitungan *output weight*. Proses alur diagram untuk menghitung *output weight* bisa dilihat di Gambar 4.14.
7. Proses perhitungan *output layer*. Proses diagram untuk menghitung *output layer* bisa dilihat di Gambar 4.15.

Pada Gambar 4.5 tersebut merupakan diagram alur proses input bias.

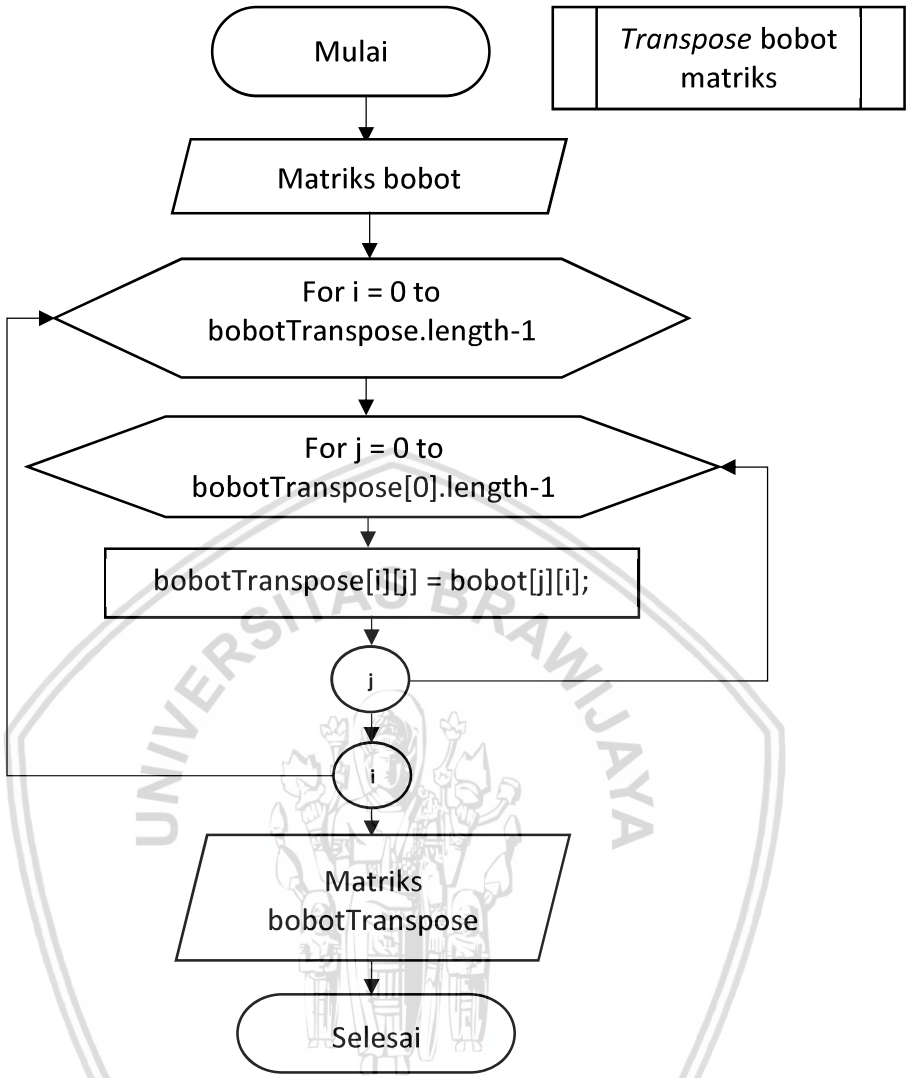


Gambar 4.5 Diagram Alur Inisialisasi Bias

Langkah diagram alur inisialisasi bias menurut dari Gambar 4.5 dapat dijelaskan:

1. Proses perulangan jumlah data latih serta *neuron hidden layer..*
2. Proses perhitungan bias secara *random* menggunakan *range* 0 sampai dengan -1.

Pada Gambar 4.6 merupakan diagram alur matriks weight *transpose.*

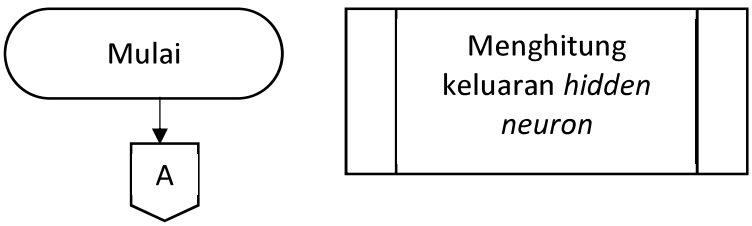


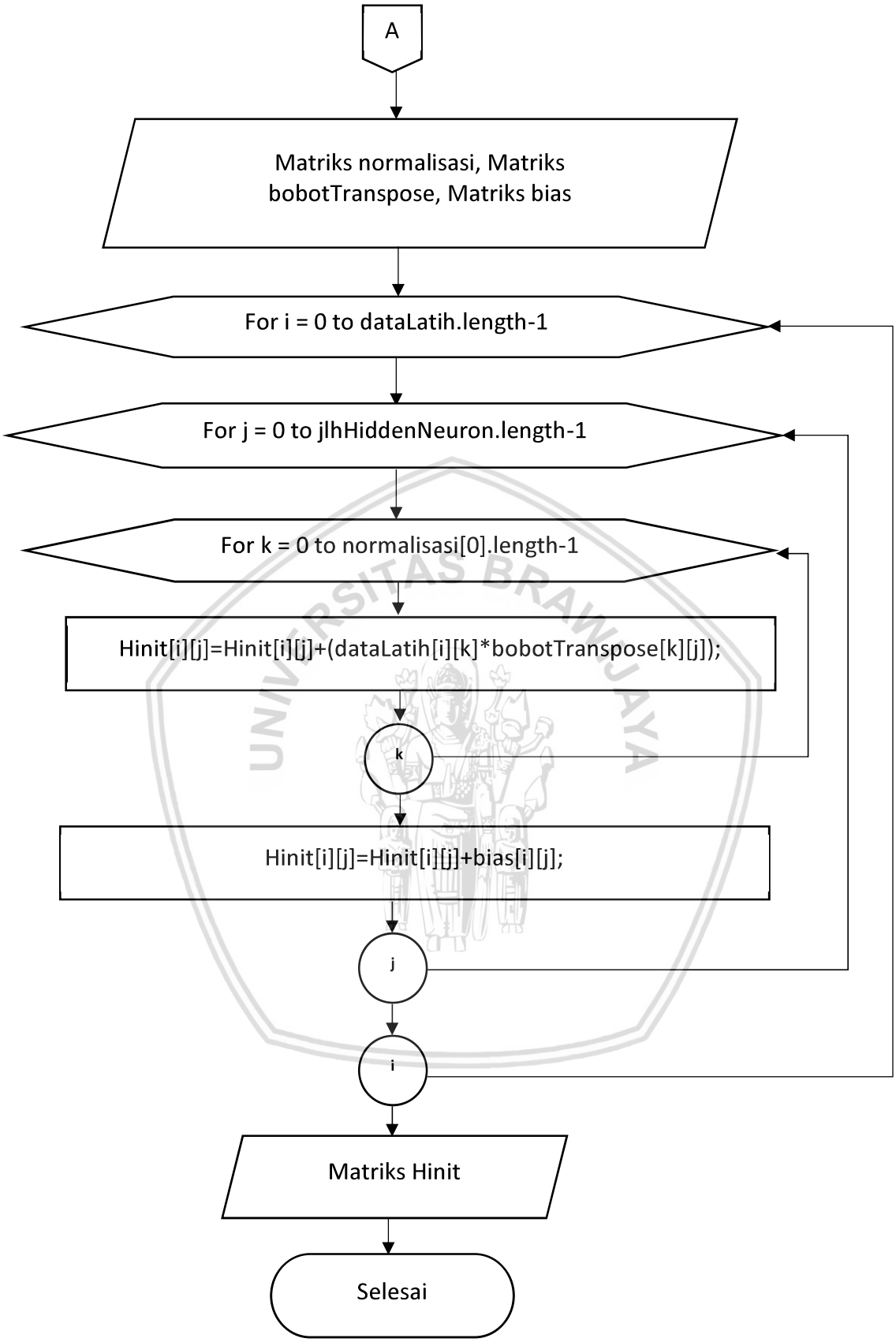
Gambar 4.6 Diagram Alur *Transpose* Matriks Bobot

Langkah diagram alur proses matriks *transpose* bobot menurut dari Gambar 4.6 dapat dijelaskan:

1. *Input* matriks *weight.*
2. *Transpose* matriks dengan mengubah nilai ordo matriks yang pertamanya baris, kolom menjadi kolom, baris.
3. *Output* sistem berupa matriks *weight transpose.*

Pada Gambar 4.7 merupakan diagram keluaran *hidden neuron.*



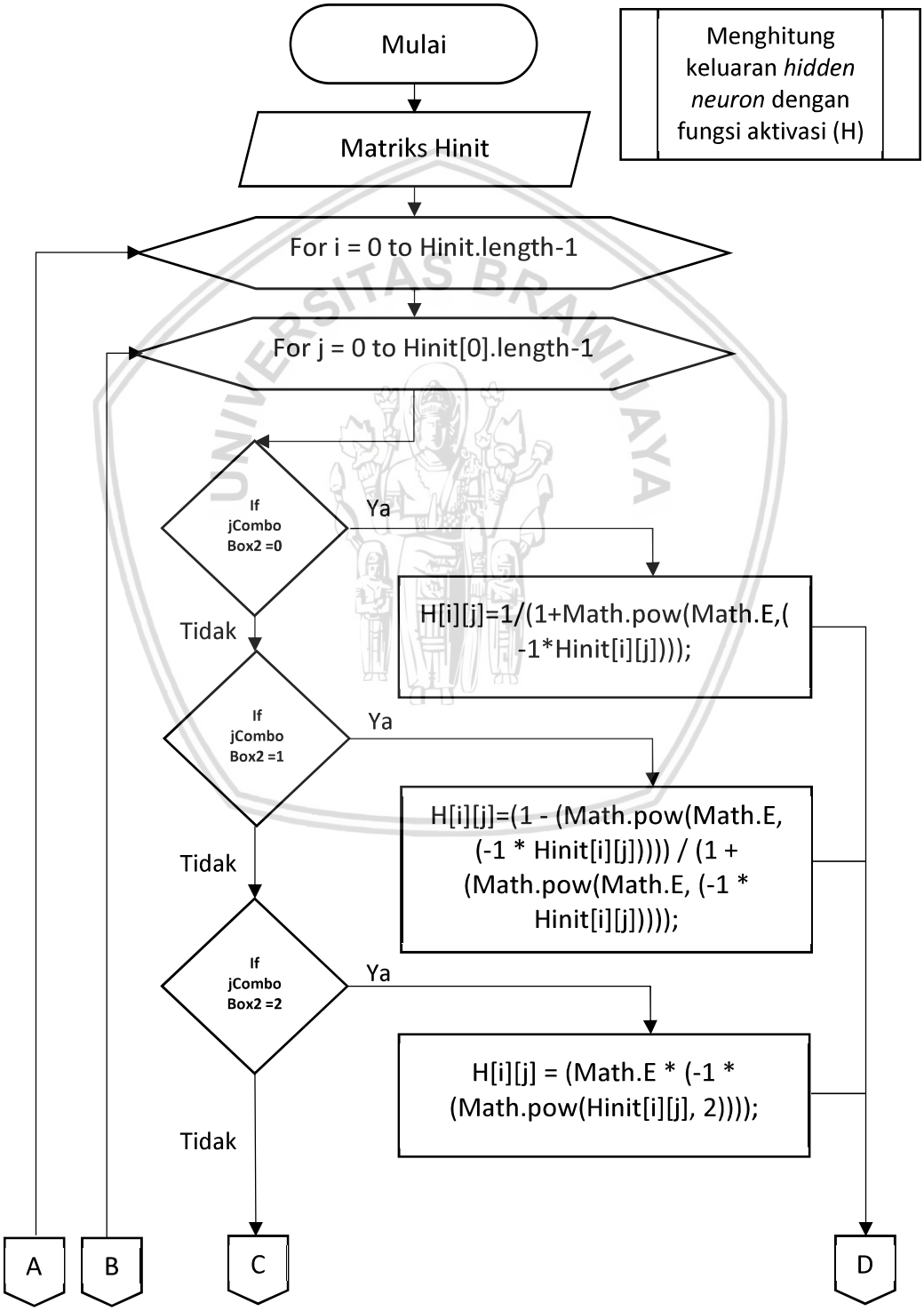


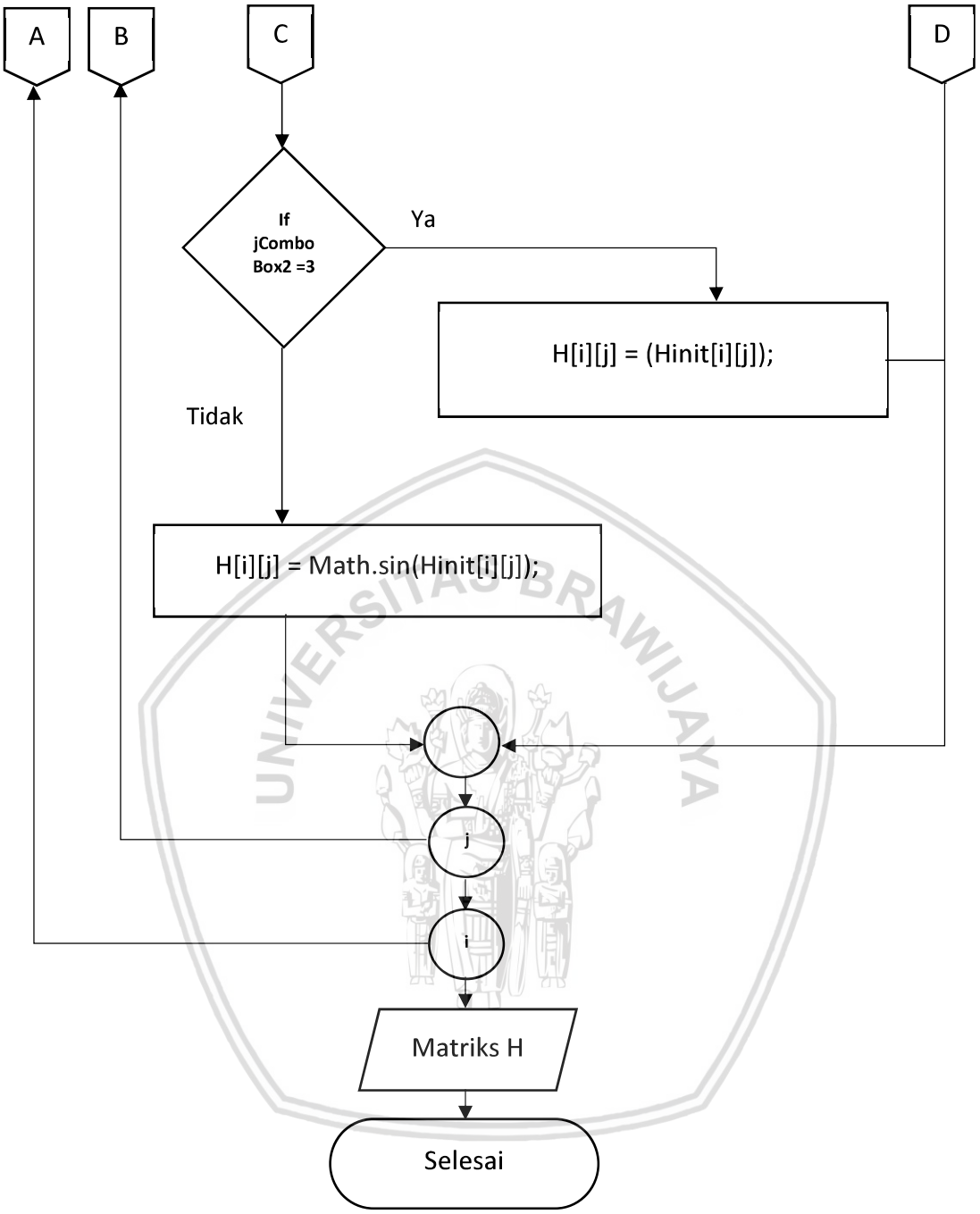
Gambar 4.7 Diagram Alur Proses Keluaran *Hidden Neuron*

Langkah diagram alur matriks *hidden neuron* menurut dari Gambar 4.7 dapat dijelaskan:

1. Sistem mendapat *input* data *training,* matriks *weight transpose* serta matriks bias.
2. Menghitung *output hidden neuron.*

Pada Gambar 4.8 merupakan diagram alur *output hidden neuron* dengan fungsi aktivasi.



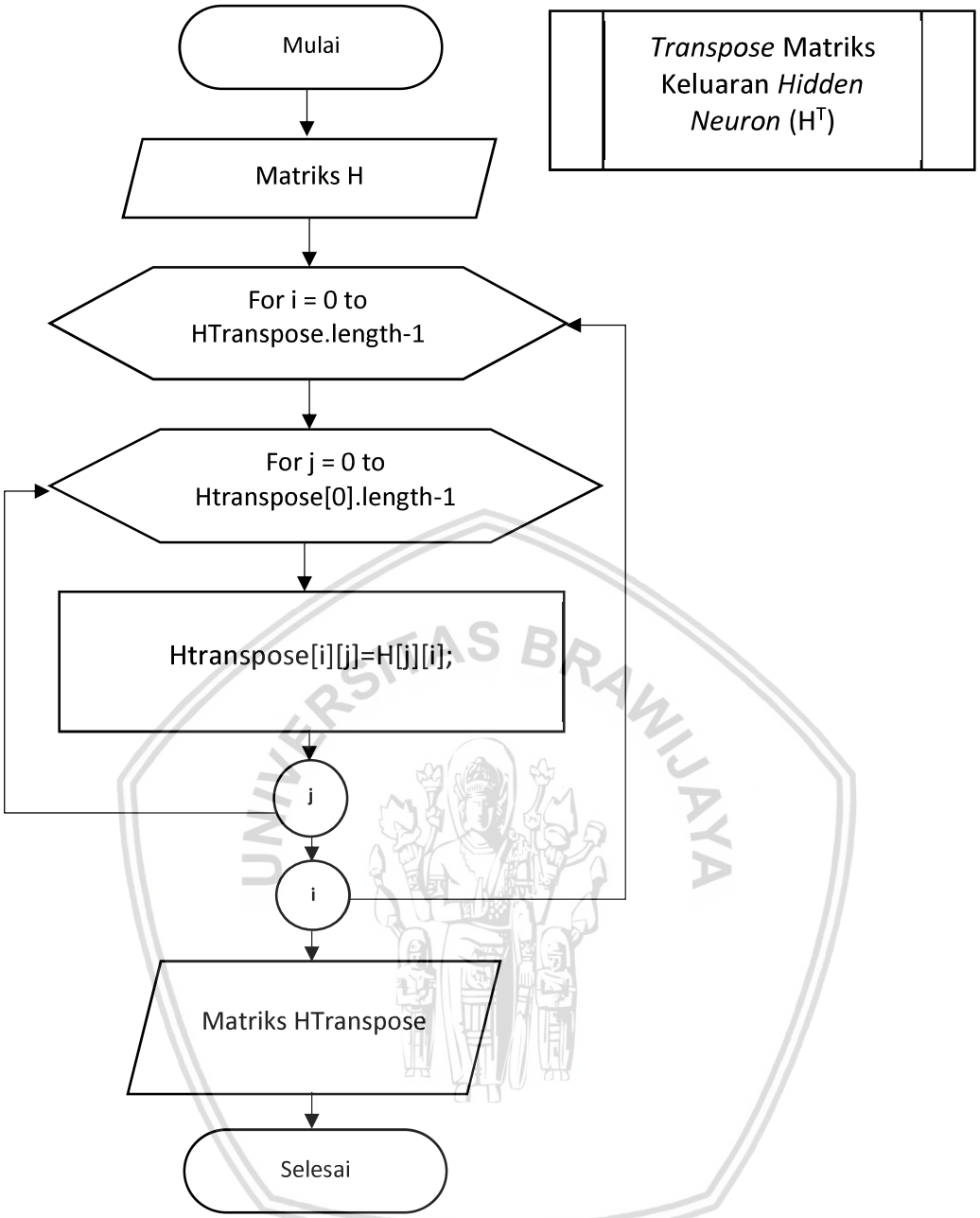


Gambar 4.8 Diagram Alur Keluaran *Hidden Neuron* dengan Fungsi Aktivasi

Langkah diagram alur *output hidden neuron* dengan fungsi aktivasi menurut dari Gambar 4.8 dapat dijelaskan:

1. Sistem mendapat *input* berupa matriks *output hidden neuron.*
2. Proses perhitungan *output hidden neuron* menggunakan fungsi aktivasi sigmoid biner.

Pada Gambar 4.9 merupakan diagram alur matriks *transpose output hidden neuron* dengan fungsi aktivasi.

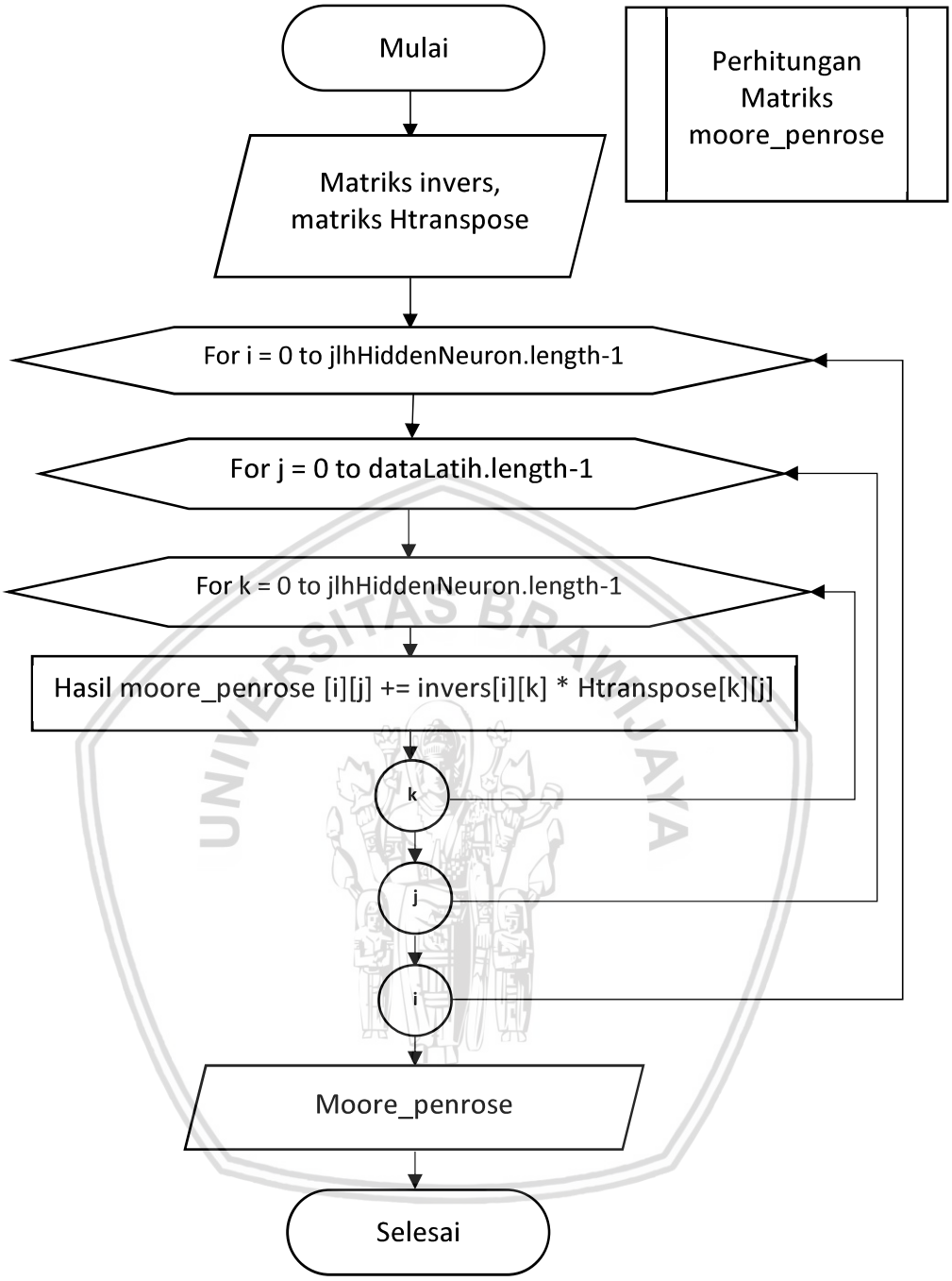


Gambar 4.9 Diagram Alur *Transpose* Matriks Keluaran *Hidden Neuron*

Langkah diagram alur matriks *transpose output hidden neuron* dengan fungsi aktivasi menurut dari Gambar 4.9 dapat dijelaskan:

1. Sistem mendapat *input* berupa matriks *output hidden neuron* dengan fungsi aktivasi.
2. Proses perhitungan matriks *transpose* dengan mengubah ordo matriks yang pertamanya baris, kolom menjadi kolom, baris.
3. Output dari sistem berupa matriks *transpose* keluaran *hidden neuron* dengan fungsi aktivasi.

Pada Gambar 4.10 merupakan diagram alur proses perhitungan moore-penrose.

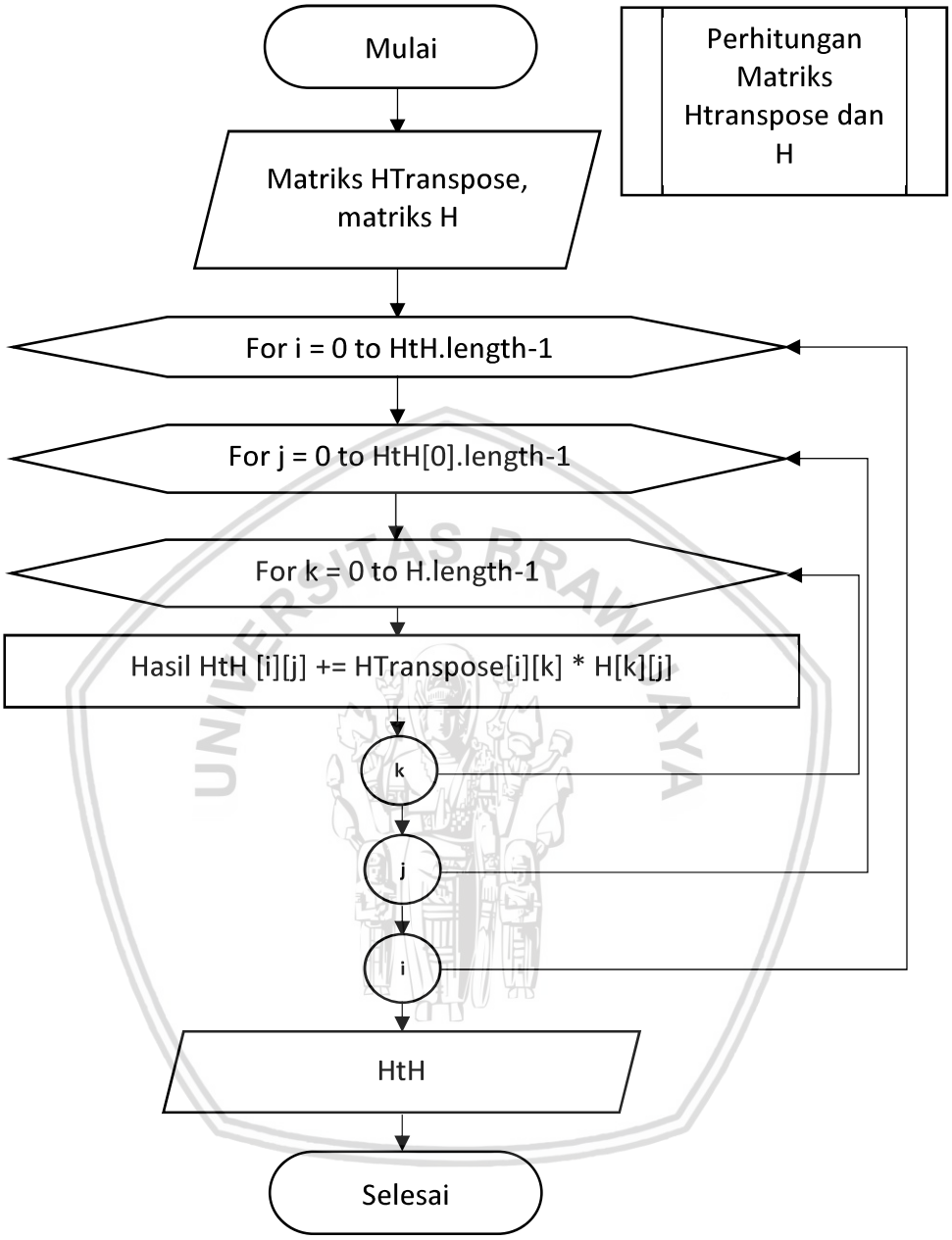


Gambar 4.10 Diagram Alur Perhitungan *Moore-Penrose*

Langkah diagram alur proses perhitungan *moore-penrose* menurut dari Gambar 4.10 dapat dijelaskan:

1. *Input* matriks dari *output hidden neuron* dengan fungsi aktivasi yang telah di *transpose* serta matriks keluaran *hidden neuron.*
2. Proses perhitungan perkalian antara matriks keluaran *hidden neuron* dengan fungsi aktivasi yang sudah di *transpose* dengan matriks *hidden neuorn.*

Pada Gambar 4.11 merupakan diagram alur proses perhitungan matriks Htranspose dan H.

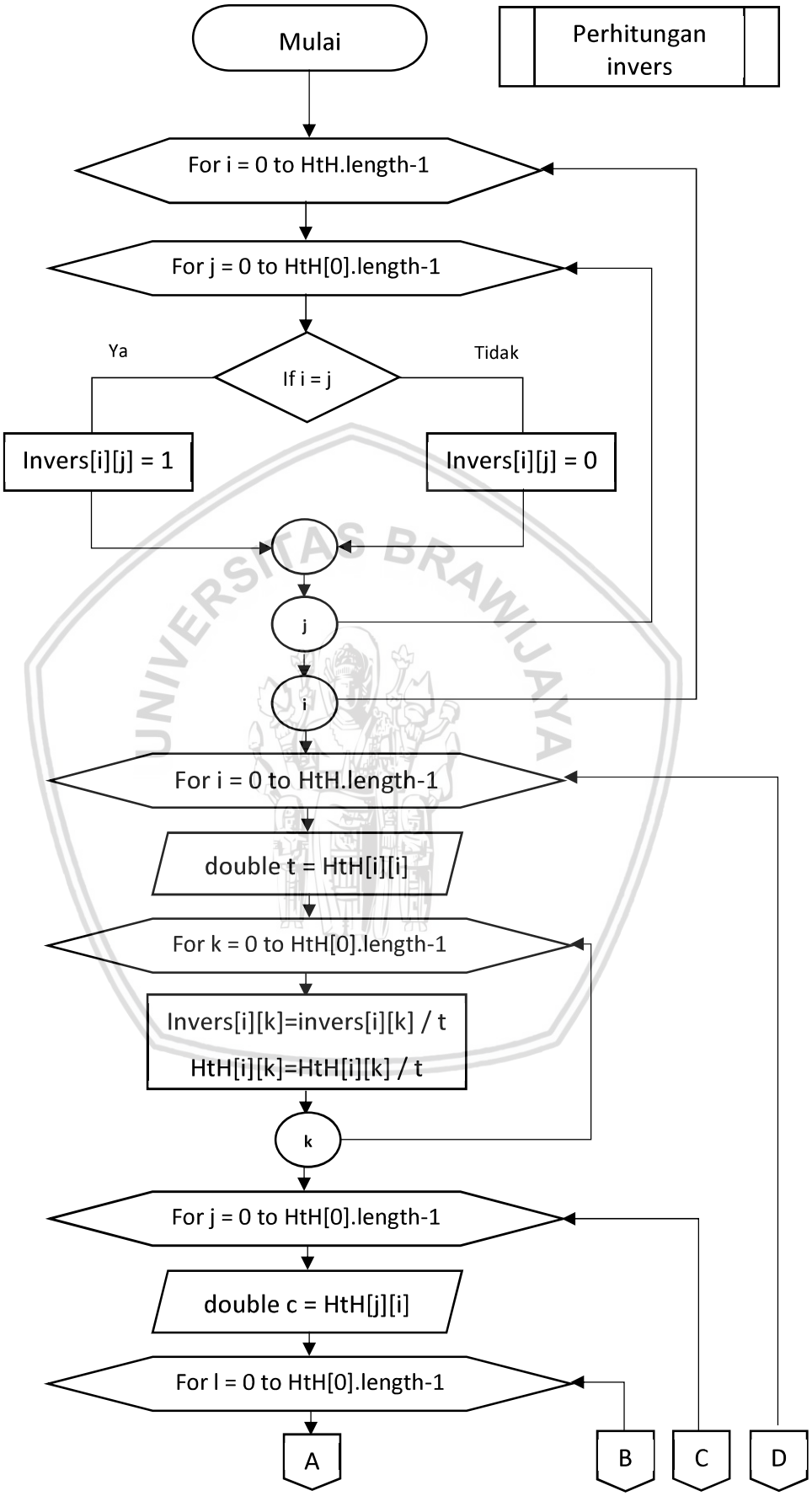


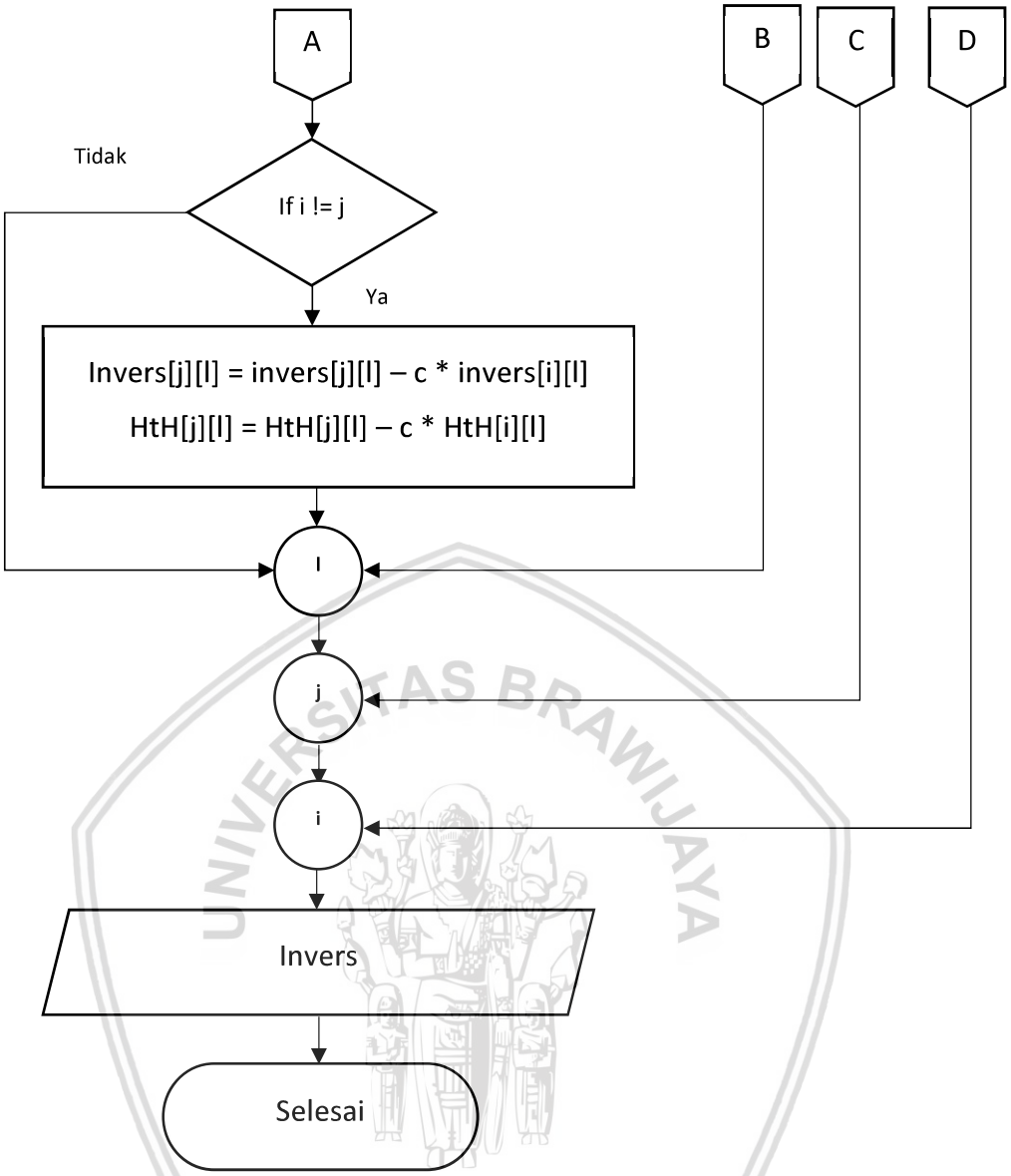
Gambar 4.11 Diagram Alur Perhitungan HtH

Langkah diagram alur proses perhitungan HtH menurut dari Gambar 4.11 dapat dijelaskan:

1. *Input* matriks dari keluaran HTranspose serta matriks keluaran H.
2. Proses perhitungan perkalian antara matriks keluaran HTanspose dengan matriks H.
3. Hasil berupa matriks keluaran HtH.

Pada Gambar 4.12 merupakan diagram alur proses perhitungan *invers*.


Mulai
Perhitungan invers
For i = 0 to HtH.length-1
For j = 0 to HtH[0].length-1
Ya
If i = j
Tidak
Invers[i][j] = 1
Invers[i][j] = 0
j
i
For i = 0 to HtH.length-1
double t = HtH[i][i]
For k = 0 to HtH[0].length-1
Invers[i][k]=invers[i][k] / t
HtH[i][k]=HtH[i][k] / t
k
For j = 0 to HtH[0].length-1
double c = HtH[j][i]
For l = 0 to HtH[0].length-1
A
B
C
D

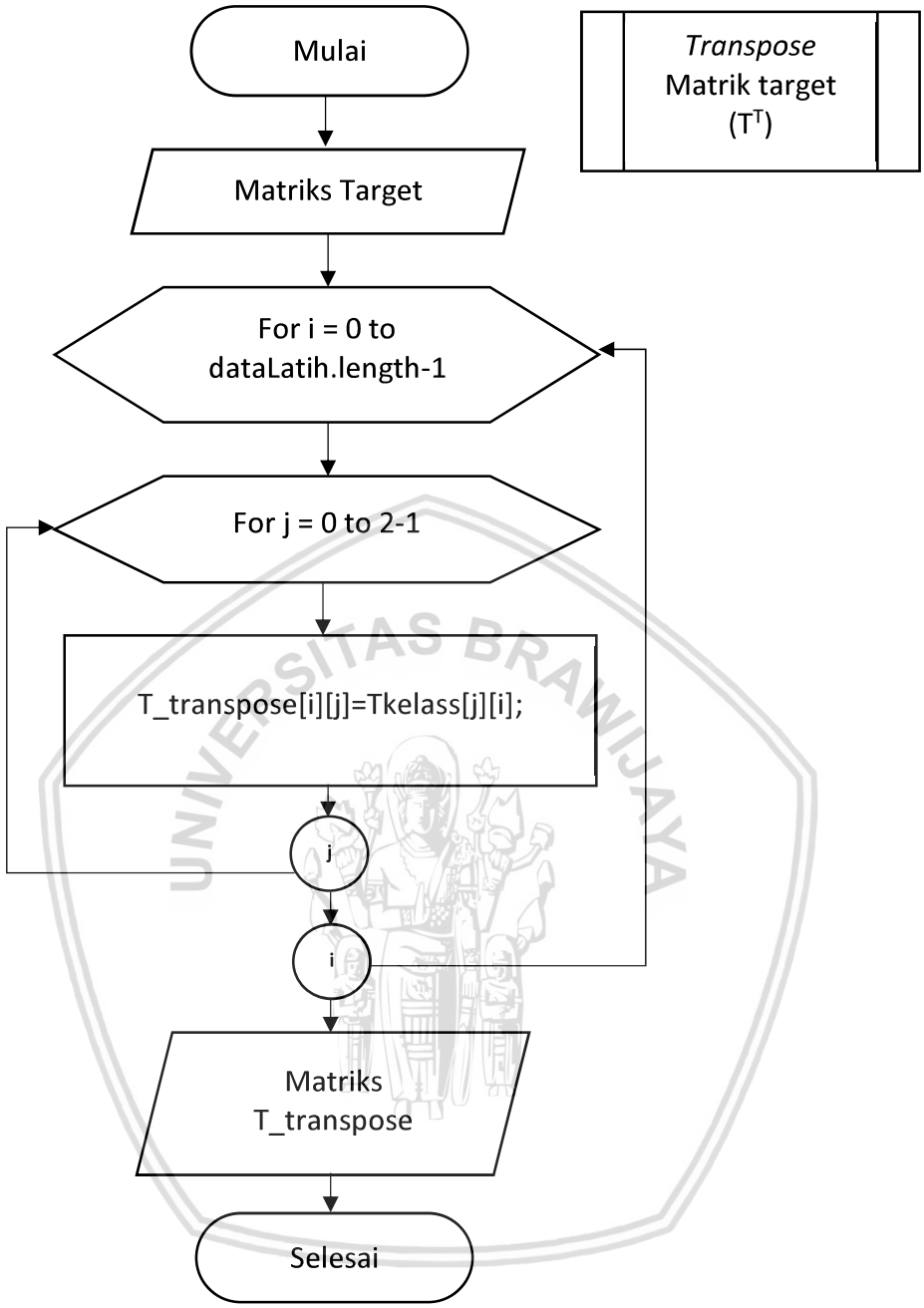


Gambar 4.12 Diagram Alur Perhitungan *Invers*

Langkah diagram alur proses perhitungan *Invers* menurut dari Gambar 4.12 dapat dijelaskan:

1. Inisialisasi matriks *invers* terlebih dahulu. Nilai dari matriks HtH dimasukan ke *variable t.*
2. Proses perhitungan matriks *invers* dengan membagi matriks *invers* dengan *variable t* serta membagi matriks HtH dengan *variabel t.* Nilai dari matriks HtH dimasukan ke *variable c.*
3. Proses perhitungan matriks *invers* dikurang dengan nilai *variable c* kemudian dikalikan dengan matriks *invers,* serta nilai matriks HtH dikurang dengan nilai *variable* c kemudian dikalikan dengan matriks HtH.
4. Hasil berupa matriks keluaran *invers.*

Pada Gambar 4.13 merupakan diagram alur matriks *transpose* target.

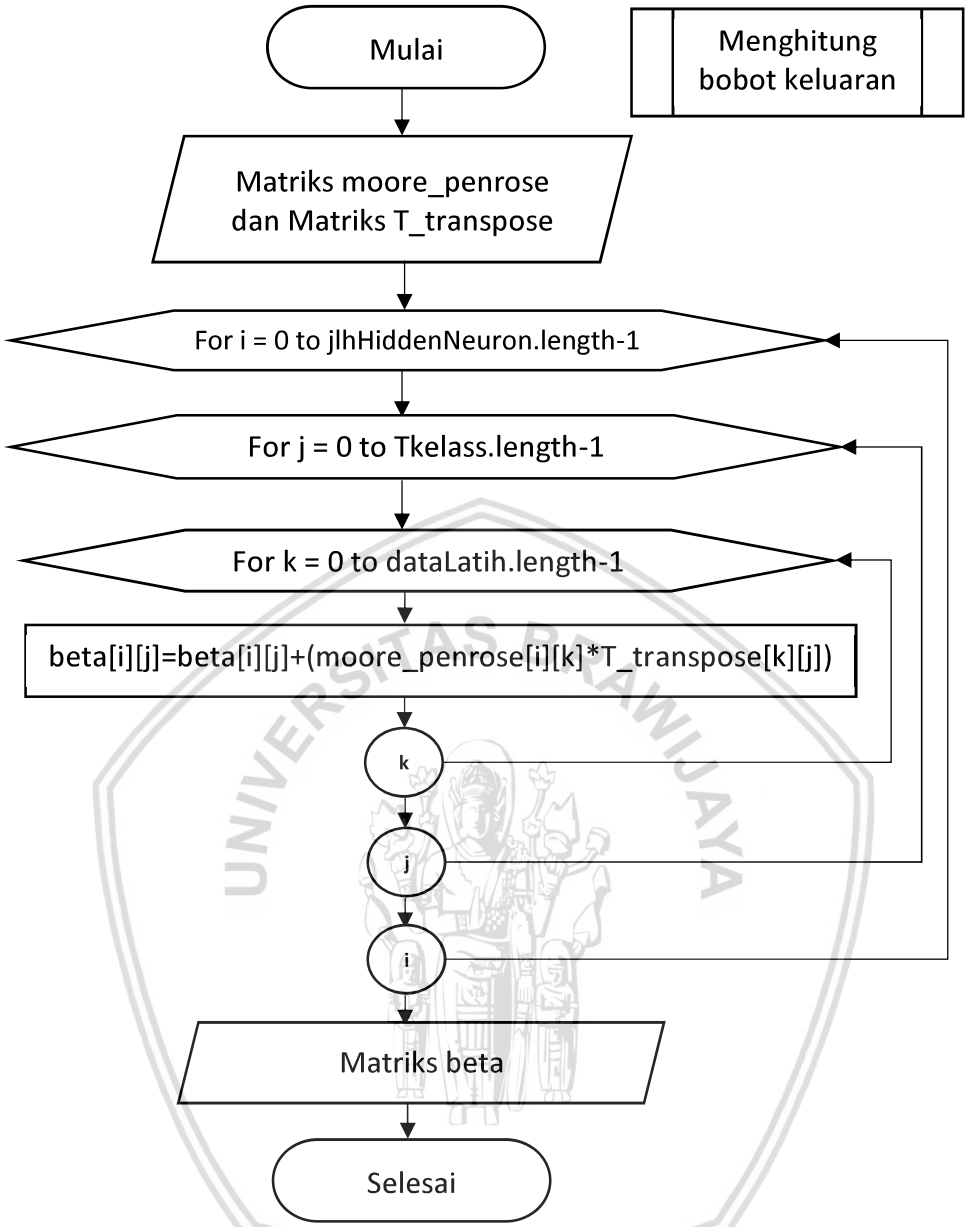


Gambar 4.13 Diagram Alur *Transpose* Matriks Target

Langkah diagram alur proses *transpose* matriks target menurut dari Gambar 4.13 dapat dijelaskan:

1. *Input* berupa nilai matriks target.
2. Proses matriks *transpose* dengan cara mengubah ordo matriks yang pertama baris, kolom menjadi kolom, baris.
3. *Output* dari sistem berupa matriks target yang sudah di *transpose*.

Pada Gambar 4.14 merupakan diagram alur proses perhitungan *output weight.*

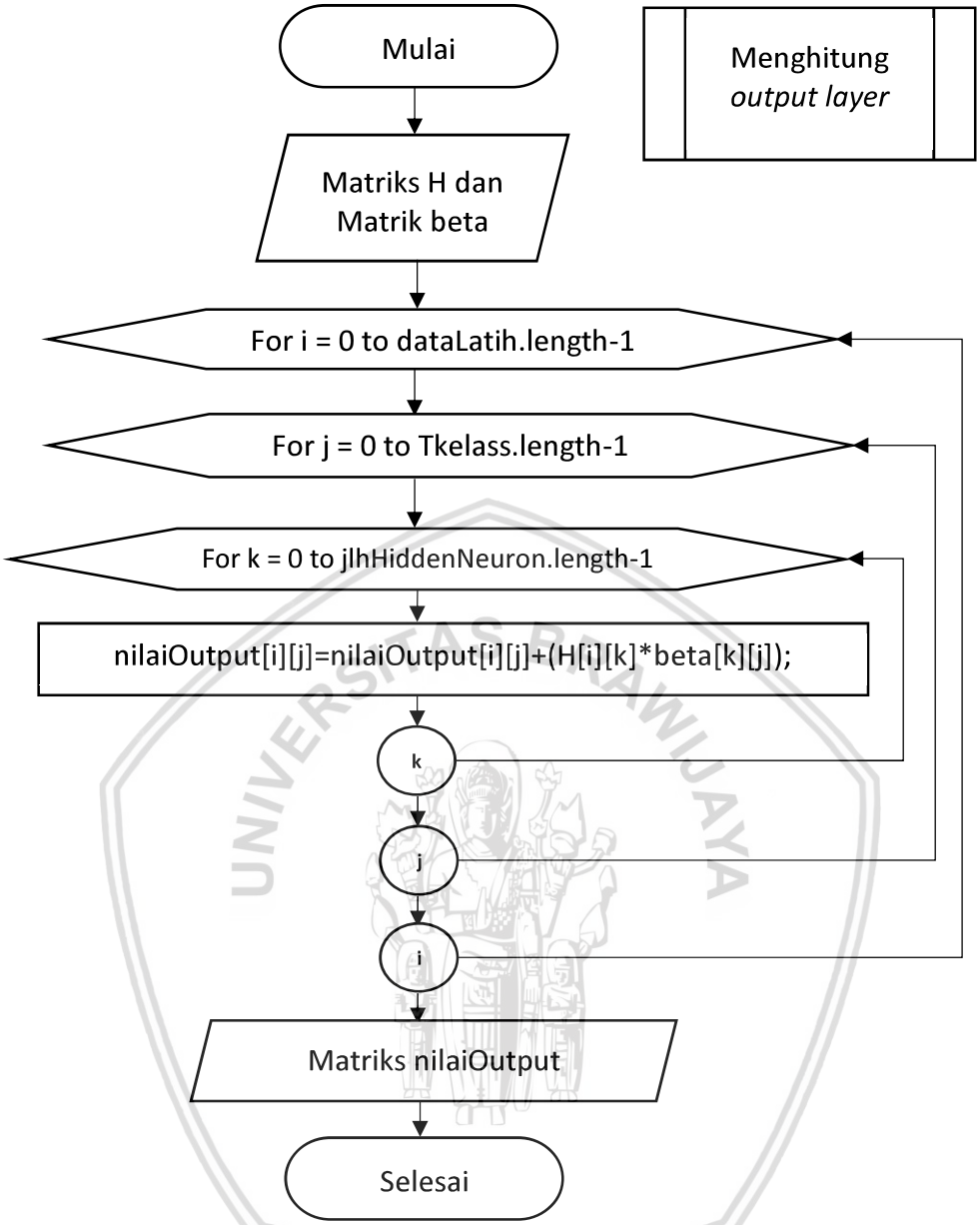


Gambar 4.14 Diagram Alur Bobot Keluaran

Langkah diagram alur proses perhitungan *output weight* menurut dari Gambar 4.14 dapat dijelaskan:

1. *Input* matrik *moore_penrose* serta matriks *transpose* target.
2. Proses perhitungan perkalian antara matriks *moore_penrose* dengan matriks *transpose* target.
3. *Output* dari sistem berupa matriks *output weight*.

Pada Gambar 4.15 merupakan diagram alur proses perhitungan *output layer.*



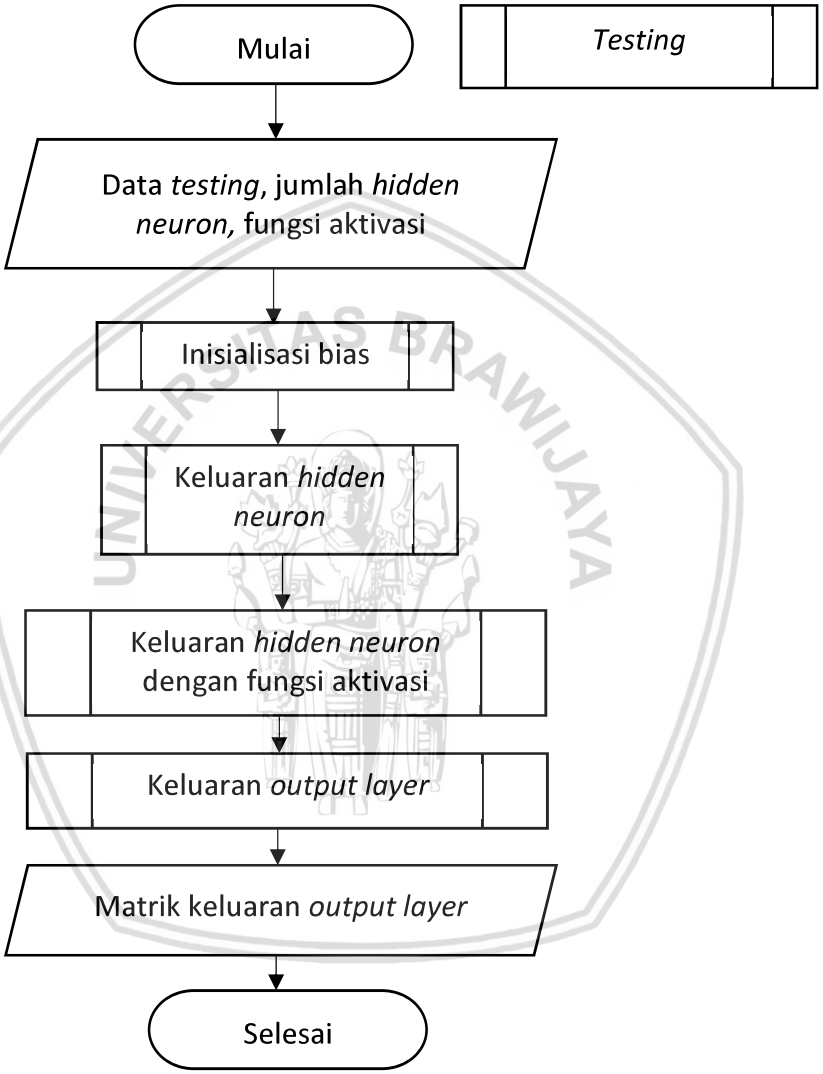
Gambar 4.15 Diagram Alur Keluaran *Output Layer*

Langkah diagram alur proses perhitungan *output layer* menurut dari Gambar 4.15 dapat dijelaskan:

1. *Input* matriks *hidden neuron* dengan fungsi aktivasi dan matriks *output weight*.
2. Proses perhitungan perkalian antara matriks *output hidden neuron* dengan fungsi aktivasi dengan matriks *output weight*.
3. *Output* dari sistem berupa matriks *output layer.*

4.2.3 Proses *Testing*

Setelah proses *training* sudah selesai dan di dapat nilai *output layer*, selanjutnya akan dilanjutkan ke proses *testing*. Untuk nilai dari *output weight* pada proses *testing* tidak perlu dihitung. Karena pada proses *testing* akan menggunakan nilai *output weight* yang didapatkan pada proses *training* untuk menghitung *output layer* di proses *testing.* Gambar 4.16 merupakan langkah-langkah diagram alur proses testing metode *Extreme Learning Machine*.



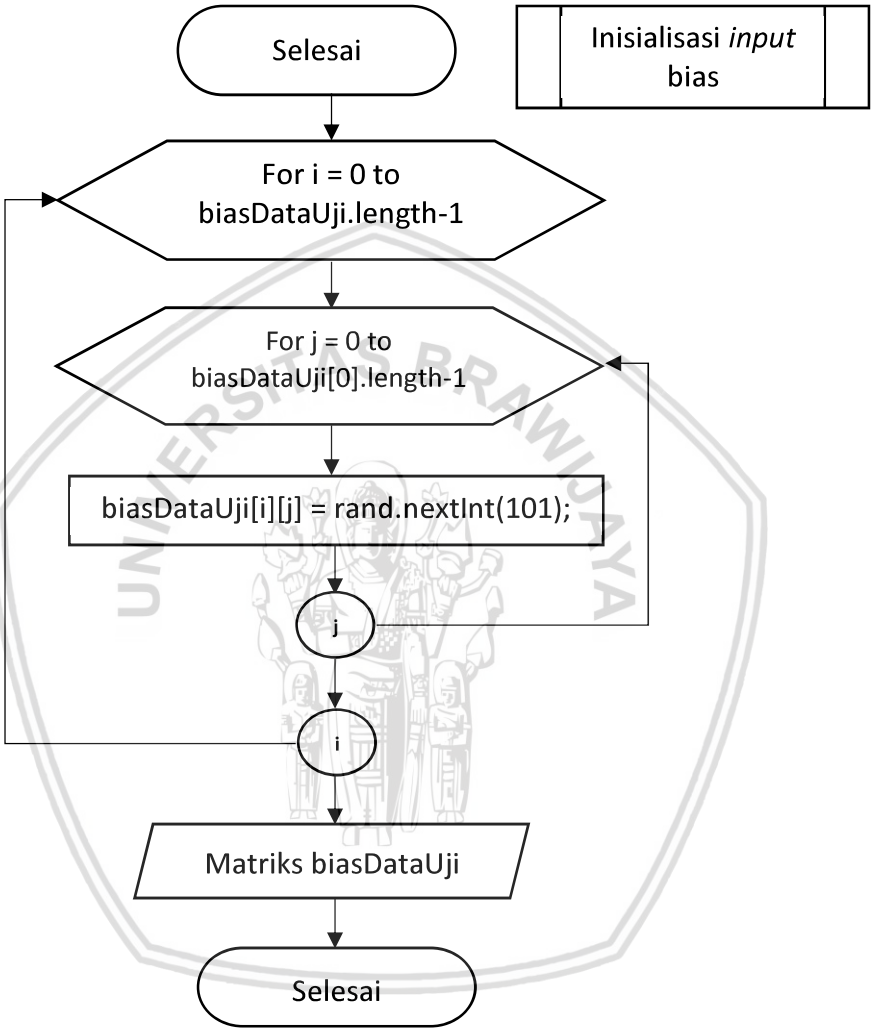
Gambar 4.16 Diagram Alur Proses *Testing*

Langkah tahapan proses *testing* algoritme Extreme Learning Machine adalah:

1. Sistem mendapat nilai *input* data *testing*, nilai *output weight* dari proses *training*.
2. Inisialisasi nilai bias menggunakan *range* nilai 0 sampai dengan 1. Proses alur diagram inisialisasi bias bisa dilihat di Gambar 4.17.
3. Proses perhitungan keluaran *neuron hidden layer.* Proses alur diagram keluaran *neuron hidden layer* bisa dilihat di Gambar 4.18.

4. Proses perhitungan keluaran *neuron hidden layer* dengan fungsi aktivasi. Proses alur diagram *neuron hidden layer* dengan fungsi aktivasi bisa dilihat di Gambar 4.19.
5. Proses perhitungan *output layer*. Proses diagram untuk menghitung *output layer* bisa dilihat di Gambar 4.20.

Pada Gambar 4.17 merupakan diagram alur proses inisialisasi bias.

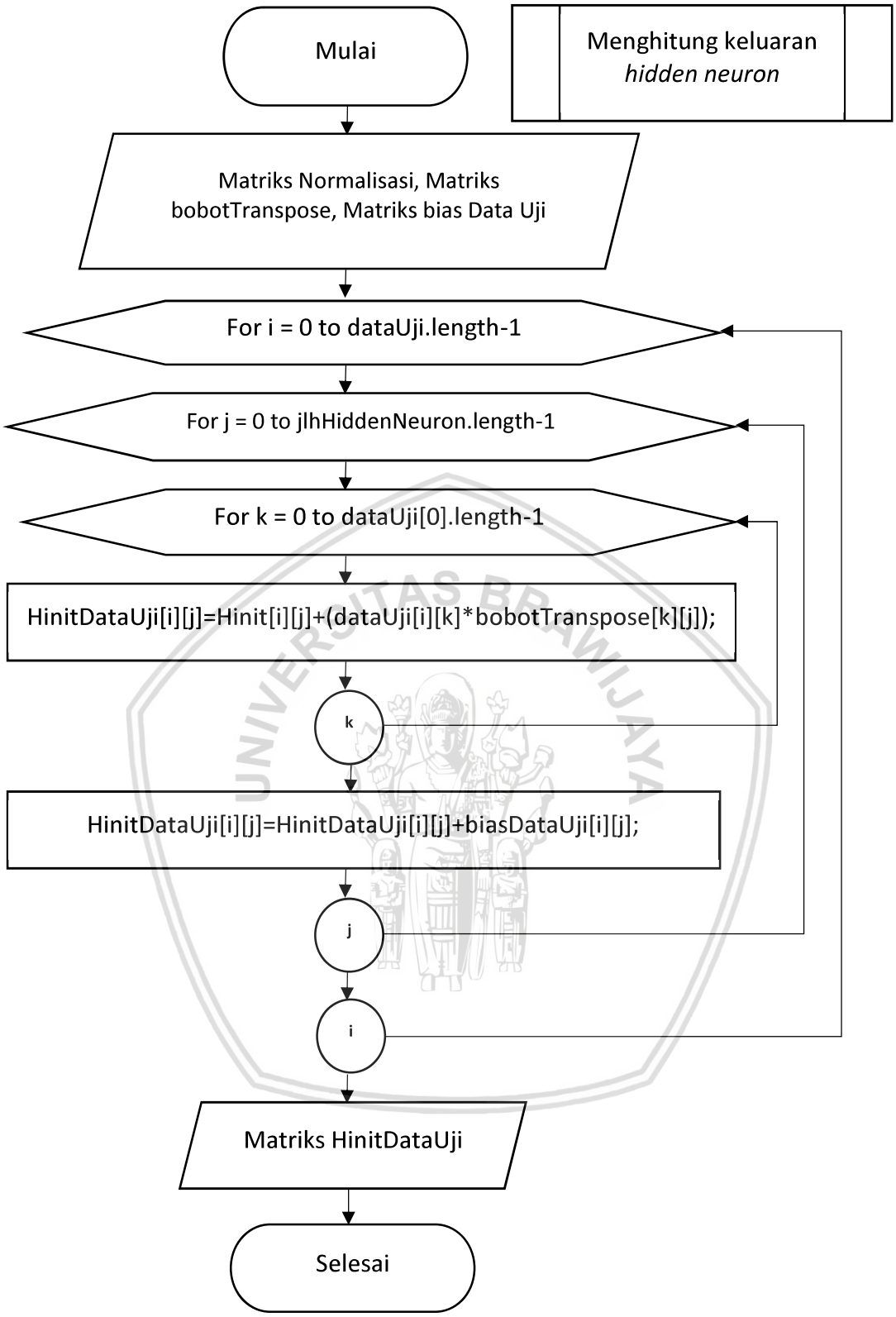


Gambar 4.17 Diagram Alur Inisialisasi Bias

Langkah diagram alur proses inisialisasi bias menurut dari Gambar 4.17 dapat dijelaskan:

1. Proses perulangan jumlah data uji serta *neuron hidden layer.*
2. Proses perhitungan bias secara *random* menggunakan *range* 0 sampai dengan -1.

Pada Gambar 4.18 merupakan diagram alur proses perhitungan *output hidden neuron.*



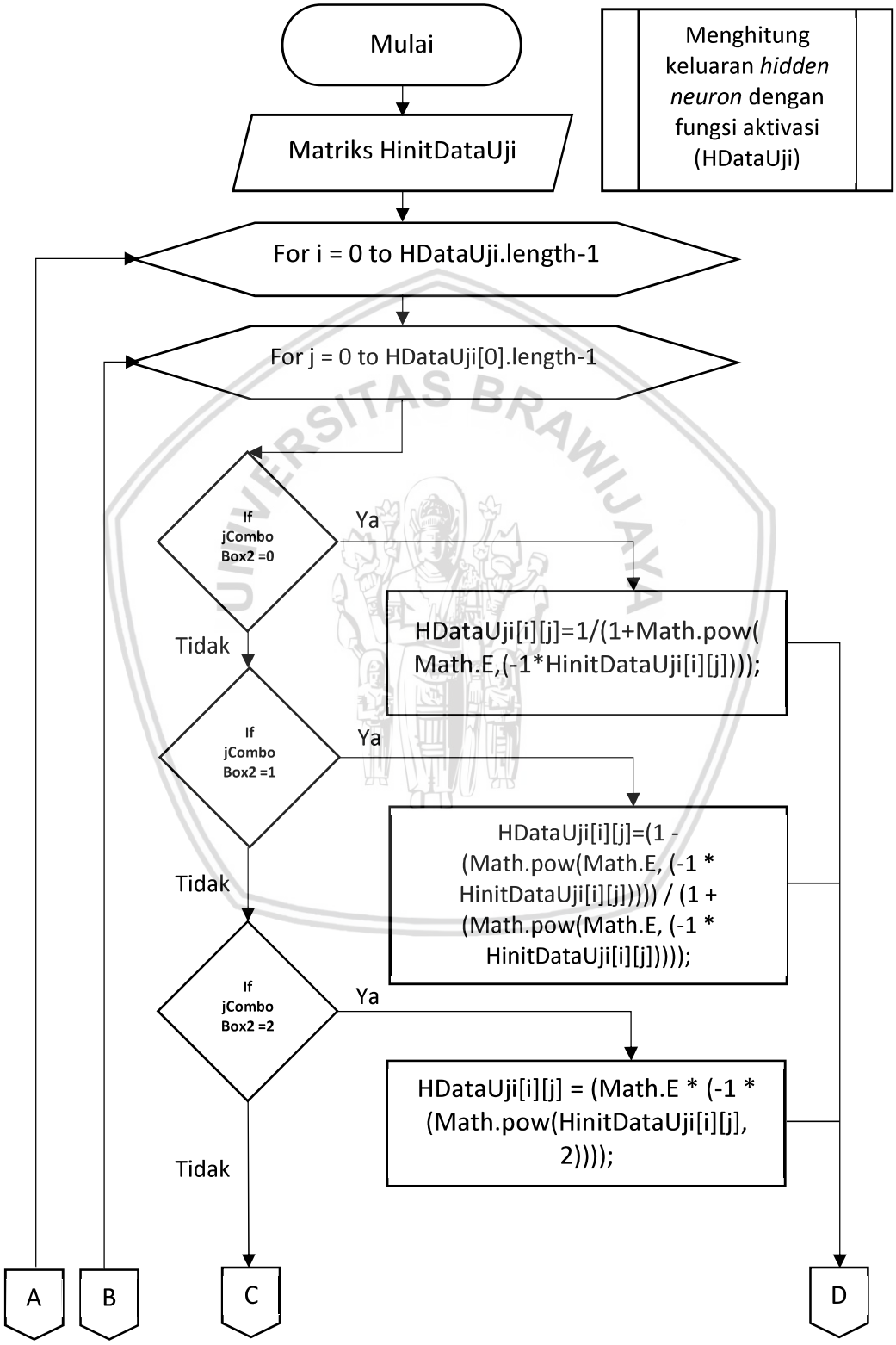
Gambar 4.18 Diagram Alur Keluran *Hidden Neuron*

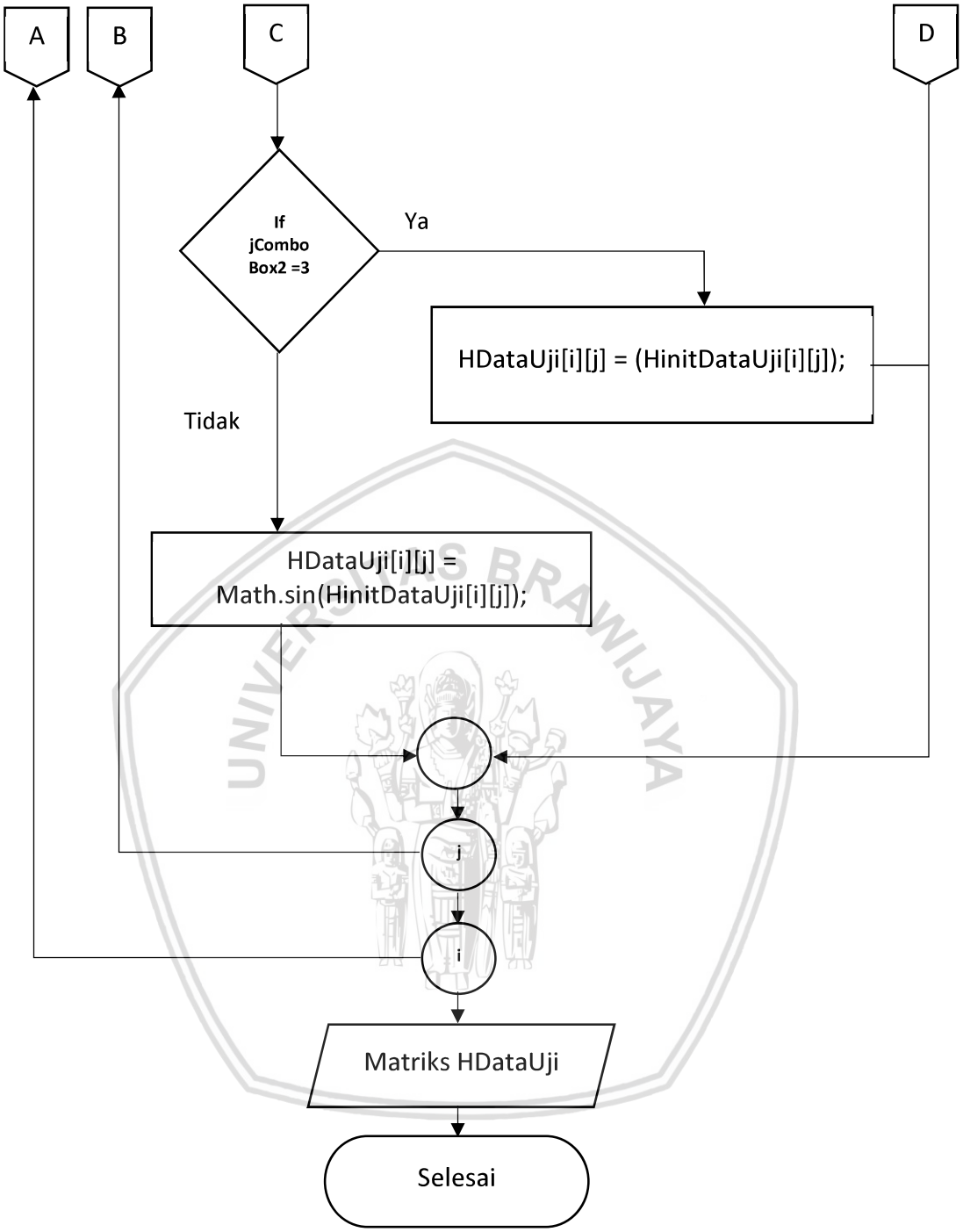
Langkah diagram alur matriks *hidden neuron* menurut dari Gambar 4.18 dapat dijelaskan:

1. Sistem mendapat *input* data *testing,* matriks *weight transpose* serta matriks bias.

2. Menghitung *output hidden neuron.*

Pada Gambar 4.19 merupakan diagram alur proses perhitungan *output hidden neuron* dengan fungsi aktivasi.



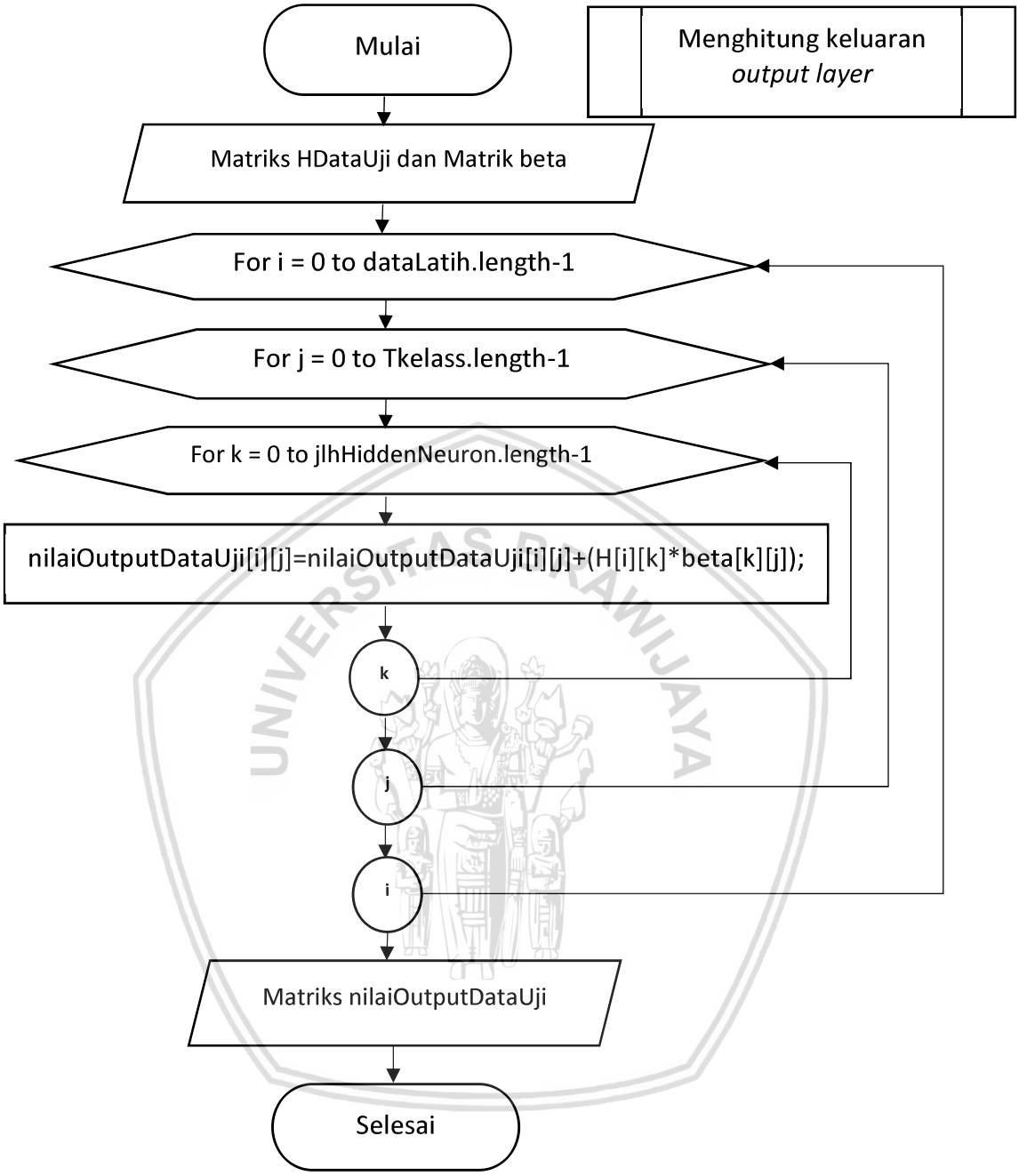


Gambar 4.19 Diagram Alur Keluaran *Hidden Neuron* dengan Fungsi Aktivasi

Langkah diagram alur *output hidden neuron* dengan fungsi aktivasi menurut dari Gambar 4.19 dapat dijelaskan:

1. Sistem mendapat *input* berupa matriks *output hidden neuron.*
2. Proses perhitungan *output hidden neuron* menggunakan fungsi aktivasi sigmoid biner.

Pada Gambar 4.20 merupakan diagram alur proses perhitungan *output layer.*



Gambar 4.20 Diagram Alur Keluaran *Output Layer*

Langkah diagram alur proses perhitungan *output layer* menurut dari Gambar 4.20 dapat dijelaskan:

1. *Input* matriks *hidden neuron* dengan fungsi aktivasi dan matriks *output weight*.
2. Proses perhitungan perkalian antara matriks *output hidden neuron* dengan fungsi aktivasi dengan matriks *output weight*.
3. *Output* dari sistem berupa matriks *output layer.*

4.3 Perhitungan Manual

Dalam penelitian ini menggunakan data untuk perhitungan manual sebanyak 25 data dengan menggunakan fungsi aktivasi sigmoid. Untuk jumlah fitur 24 fitur serta menggunakan 2 kelas yaitu kelas 1 dan kelas 2, yang mana kelas 1 mewakili penderita gagal ginjal kronis dan kelas 2 mewakili tidak menderita gagal ginjal kronis. Untuk nilai target tergantung dari kelasnya, apabila kelasnya 1 maka nilai target 1 dan -1 sebaliknya apabila kelasnya 2 maka nilai target -1 dan 1. Dalam pembagian data untuk perhitungan manual yaitu 80% digunakan sebagai data *training* dan 20% akan digunakan sebagai data *testing*. Jadi secara otomatis 20 data digunakan untuk proses *training* dan 5 data digunakan untuk proses *testing*.

1. Inisialisasi fitur

Tabel 4.1 adalah inisialisasi fitur dalam proses perhitungan manual. Nantinya terdapat 24 fitur. Data yang digunakan dalam inisialisasi perhitungan manual ini berjumlah 25 data. Tabel 4.1 merupakan data set inisialisasi perhitungan manual.

Tabel 4.1 Data Set Inisialisasi Perhitungan Manual

No	X1	X2	X3	X4	X5	...	X20	X21	X22	X23	X24	target		kls
1	55	80	1010	3	1	...	1	2	1	1	2	1	-1	1
2	40	70	1015	3	4	...	1	2	2	1	2	1	-1	1
3	52	90	1015	4	3	...	1	2	1	2	1	1	-1	1
4	48	70	1005	4	0	...	2	2	2	1	1	1	-1	1
5	6	60	1010	4	0	...	2	2	2	2	2	1	-1	1
6	61	80	1015	2	0	...	1	1	2	1	1	1	-1	1
7	60	60	1010	3	1	...	2	2	2	2	1	1	-1	1
8	48	80	1025	4	0	...	2	2	1	2	1	1	-1	1
9	59	70	1010	1	3	...	1	1	1	2	2	1	-1	1
10	35	60	1025	0	0	...	2	2	1	2	2	-1	1	2
11	61	70	1025	0	0	...	2	2	1	2	2	-1	1	2
12	62	80	1025	0	0	...	2	2	1	2	2	-1	1	2
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
16	48	60	1020	0	0	...	2	2	1	2	2	-1	1	2
17	52	80	1025	0	0	...	2	2	1	2	2	-1	1	2
18	58	70	1020	0	0	...	2	2	1	2	2	-1	1	2
19	46	60	1020	0	0	...	2	2	1	2	2	-1	1	2
20	60	80	1025	0	0	...	2	2	1	2	2	-1	1	2

Tabel 4.1 Data Set Inisialisasi Perhitungan Manual (lanjutan)

No	X1	X2	X3	X4	X5	...	X20	X21	X22	X23	X24	target		kls
21	66	70	1025	0	0	...	2	2	1	2	2	-1	1	2
22	74	60	1020	0	0	...	2	2	1	2	2	-1	1	2
23	33	80	1025	0	0	...	2	2	1	2	2	-1	1	2
24	30	80	1020	0	0	...	2	2	1	2	2	-1	1	2
25	71	70	1020	0	0	...	2	2	1	2	2	-1	1	2

Keterangan:

X1 = fitur ke 1
X2 = fitur ke 2
X3 = fitur ke 3
X4 = fitur ke 4
X5 = fitur ke 5
X6 = fitur ke 6
X7 = fitur ke 7
X8 = fitur ke 8
X9 = fitur ke 9
X10 = fitur ke 10
X11 = fitur ke 11
X12 = fitur ke 12
X13 = fitur ke 13
X14 = fitur ke 14
X15 = fitur ke 15
X16 = fitur ke 16
X17 = fitur ke 17
X18 = fitur ke 18
X19 = fitur ke 19
X20 = fitur ke 20
X21 = fitur ke 21
X22 = fitur ke 22
X23 = fitur ke 23
X24 = fitur ke 24

Data yang akan digunakan berjumlah 25 data. 20 data untuk proses *training* serta 5 data untuk proses *testing.* Dalam manualisasi ini data nomor 1 sampai dengan nomor 20 merupakan data *training,* sedangkan nomor 21 sampai dengan nomor 25 merupakan data *testing.* Tabel 4.2 serta 4.3 adalah data *training* dan data *testing.*

Tabel 4.2 Data *Training*

No	X1	X2	X3	X4	X5	X6	X7	...	X20	X21	X22	X23	X24
1	55	80	1010	3	1	1	2	...	1	2	1	1	2
2	40	70	1015	3	4	1	1	...	1	2	2	1	2
3	52	90	1015	4	3	1	2	...	1	2	1	2	1
4	48	70	1005	4	0	1	2	...	2	2	2	1	1
5	6	60	1010	4	0	2	2	...	2	2	2	2	2
6	61	80	1015	2	0	2	2	...	1	1	2	1	1
7	60	60	1010	3	1	1	2	...	2	2	2	2	1
8	48	80	1025	4	0	1	2	...	2	2	1	2	1
9	59	70	1010	1	3	2	2	...	1	1	1	2	2

Tabel 4.2 Data *Training* (lanjutan)

No	X1	X2	X3	X4	X5	X6	X7	...	X20	X21	X22	X23	X24
10	35	60	1025	0	0	1	1	...	2	2	1	2	2
11	61	70	1025	0	0	1	1	...	2	2	1	2	2
12	62	80	1025	0	0	1	1	...	2	2	1	2	2
13	34	60	1020	0	0	1	1	...	2	2	1	2	2
14	42	70	1025	0	0	1	1	...	2	2	1	2	2
15	79	80	1025	0	0	1	1	...	2	2	1	2	2
16	48	60	1020	0	0	1	1	...	2	2	1	2	2
17	52	80	1025	0	0	1	1	...	2	2	1	2	2
18	58	70	1020	0	0	1	1	...	2	2	1	2	2
19	46	60	1020	0	0	1	1	...	2	2	1	2	2
20	60	80	1025	0	0	1	1	...	2	2	1	2	2

Tabel 4.3 Data *Testing*

No	X1	X2	X3	X4	X5	X6	X7	...	X20	X21	X22	X23	X24
21	66	70	1025	0	0	1	1	...	1	1	1	2	2
22	74	60	1020	0	0	1	1	...	2	2	1	2	2
23	33	80	1025	0	0	1	1	...	2	2	1	2	2
24	30	80	1020	0	0	1	1	...	2	2	1	2	2
25	71	70	1020	0	0	1	1	...	2	2	1	2	2

Untuk contoh perhitungan manual pada penelitian ini akan menggunakan *neuron hidden layer* sebanyak 2 serta menggunakan fungsi aktivasi *sigmoid*.

2. Proses normalisasi data

Pada tahap normalisasi akan menggunakan *MinMax Normalization.* Dari ke 25 data untuk perhitungan manual akan dicari nilai yang minimum dan maksimum. Untuk nilai batasan nilai maksimum dan minimum di dapat dari wawancara dengan pakar. Pada Tabel 4.4 merupakan nilai minimum dan nilai maksimum yang di dapat dari setiap masing-masing fitur.

Tabel 4.4 Nilai Minimum Dan Maksimum Dari Setiap Fitur

Fitur 1	Min	Max
	6	79
Fitur 2	Min	Max
	60	90

Tabel 4.4 Nilai Minimum Dan Maksimum Dari Setiap Fitur (lanjutan)

Fitur 3	Min	Max
	1005	1025
Fitur 4	Min	Max
	0	4
Fitur 5	Min	Max
	0	4
Fitur 6	Min	Max
	1	2
Fitur 7	Min	Max
	1	2
Fitur 8	Min	Max
	1	2
Fitur 9	Min	Max
	1	2
Fitur 10	Min	Max
	75	424
Fitur 11	Min	Max
	15	166
Fitur 12	Min	Max
	0.5	11.9
Fitur 13	Min	Max
	111	147
Fitur 14	Min	Max
	2.5	47
Fitur 15	Min	Max
	7.7	17.2
Fitur 16	Min	Max
	23	54
Fitur 17	Min	Max
	5000	16700

Tabel 4.4 Nilai Minimum Dan Maksimum Dari Setiap Fitur (lanjutan)

Fitur 18	Min	Max
	2.9	6.4
Fitur 19	Min	Max
	1	2
Fitur 20	Min	Max
	1	2
Fitur 21	Min	Max
	1	2
Fitur 22	Min	Max
	1	2
Fitur 23	Min	Max
	1	2
Fitur 24	Min	Max
	1	2

Setelah mendapat nilai minimal dan maksimal kemudian langsung melakukan proses perhitungan normalisasi dengan *MinMax Normalization.* Contoh proses perhitungan manual untuk normalisasi.

$$d_{1,1} = \frac{d_{1,1} - min}{\max - min} = \frac{55-6}{79-6} = 0{,}671233$$

Tabel 4.5 merupakan hasil dari proses normalisasi dari keseluruhan data.

Tabel 4.5 Normalisasi Data

No	X1	X2	X3	X4	X5	...	X20	X21	X22	X23	X24
1	0,671233	0,666667	0,25	0,75	0,25	...	0	1	0	0	1
2	0,465753	0,333333	0,5	0,75	1	...	0	1	1	0	1
3	0,630137	1	0,5	1	0,75	...	0	1	0	1	0
4	0,575342	0,333333	0	1	0	...	1	1	1	0	0
5	0	0	0,25	1	0	...	1	1	1	1	1
6	0,753425	0,666667	0,5	0,5	0	...	0	0	1	0	0
7	0,739726	0	0,25	0,75	0,25	...	1	1	1	1	0
8	0,575342	0,666667	1	1	0	...	1	1	0	1	0
9	0,726027	0,333333	0,25	0,25	0,75	...	0	0	0	1	1

Tabel 4.5 Normalisasi Data (lanjutan)

No	X1	X2	X3	X4	X5	...	X20	X21	X22	X23	X24
10	0,39726	0	1	0	0	...	1	1	0	1	1
11	0,753425	0,333333	1	0	0	...	1	1	0	1	1
12	0,767123	0,666667	1	0	0	...	1	1	0	1	1
13	0,383562	0	0,75	0	0	...	1	1	0	1	1
14	0,493151	0,333333	1	0	0	...	1	1	0	1	1
15	1	0,666667	1	0	0	...	1	1	0	1	1
16	0,575342	0	0,75	0	0	...	1	1	0	1	1
17	0,630137	0,666667	1	0	0	...	1	1	0	1	1
18	0,712329	0,333333	0,75	0	0	...	1	1	0	1	1
19	0,547945	0	0,75	0	0	...	1	1	0	1	1
20	0,739726	0,666667	1	0	0	...	1	1	0	1	1
21	0,821918	0,333333	1	0	0	...	1	1	0	1	1
22	0,931507	0	0,75	0	0	...	1	1	0	1	1
23	0,369863	0,666667	1	0	0	...	1	1	0	1	1
24	0,328767	0,666667	0,75	0	0	...	1	1	0	1	1
25	0,890411	0,333333	0,75	0	0	...	1	1	0	1	1

3. Inisialisasi *input weight* dan bias

Pemilihan secara *random input weight* dengan nilai *range* -1 sampai dengan 1 serta nilai *input* bias dengan nilai *range* 0 sampai dengan 1. Sedangkan *neuron hidden layer* yang digunakan berjumlah 2. Tabel 4.6 merupakan matriks *input weight.*

Tabel 4.6 Matriks Nilai *Input Weight*

No	X1	X2	X3	X4	X5	...	X20	X21	X22	X23	X24
1	0,17	0,22	0,48	0,67	0,68	...	0,18	0,50	0,35	1,00	0,89
2	0,37	0,16	0,43	0,08	0,64	...	0,33	1,00	0,75	0,61	0,73

Matriks pada *input weight* di *transpose* menjadi matriks baru. Tabel 4.7 merupakan matriks *input weight* yang sudah dilakukan proses *transpose.*

Tabel 4.7 Matriks *Transpose Input Weight*

W	1	2
1	0,17	0,37
2	0,22	0,16

Tabel 4.7 Matriks *Transpose Input Weight* (lanjutan)

W	1	2
3	0,48	0,43
4	0,67	0,08
5	0,68	0,64
6	0,63	0,74
7	0,42	0,93
8	0,15	0,81
9	0,40	0,78
10	0,86	0,81
11	0,92	0,83
12	0,57	0,27
13	1,00	0,52
14	0,50	0,81
15	0,52	0,08
16	0,08	0,99
17	0,99	0,56
18	0,93	0,33
19	0,93	0,05
20	0,18	0,33
21	0,50	1,00
22	0,35	0,75
23	1,00	0,61
24	0,89	0,73

Hidden neuron yang digunakan dalam manualisasi ini berjumlah dua. Tabel 4.8 merupakan nilai matriks bias.

Tabel 4.8 Nilai Matriks Bias

No	1	2
1	0,19	0,74
2	0,90	0,10
3	0,96	0,79
4	0,29	0,21

Tabel 4.8 Nilai Matriks Bias (lanjutan)

No	1	2
5	0,14	0,25
6	0,54	0,23
7	0,56	0,74
8	0,35	0,22
9	0,27	0,14
10	0,23	0,45
11	0,51	0,43
12	0,55	0,90
13	0,34	0,86
14	0,38	0,29
15	0,98	0,92
16	0,17	0,98
17	0,05	0,20
18	0,37	0,55
19	0,25	0,30
20	0,22	0,27

4.3.2 Perhitungan Manual Proses *Training*

Dalam penelitian ini menggunakan fungsi aktivasi sigmoid, data yang digunakan dalam perhitungan manual proses *training* berjumlah 20 data dan juga dalam penelitian ini menggunakan jumlah neuron pada *hidden layer* sebanyak 2. Data ini didapatkan dari presentase antara data untuk *training* serta data untuk *testing* yaitu 80% data untuk *training* serta 20% data untuk *testing* dari jumlah keseluruhan data dalam perhitungan manual sebanyak 25 data. Tahapan proses *training* algoritma *Extreme Learning Machine*:

Langkah pertama yaitu hitung nilai matriks keluaran hidden layer (*Hinit*). Contoh proses perhitungan dari *hidden layer.*

$$H_{init} = (X_{train} * W^T) + b$$

H_{init} = ((0,671233*0,17)+(0,666667*0,22)+(0,25*0,48)+(0,75* 0,67)+(0,25* 0,68)+(0* 0,63)+(1* 0,42)+(0* 0,15)+(0* 0,40)+(0,398281* 0,86)+(0,384* 0,92)+(0,298*0,57)+(0,722* 1,00)+(0,054*0,50)+(0,337* 0,52)+(0,355* 0,08)+(0,205*0,99)+(0,229* 0,93)+(0* 0,93)+(0* 0,18)+(1* 0,50)+(0*0,35)+(0*1,00)+(1*0,89) +0,19)= 5,280182

Tabel 4.9 adalah hasil dari proses perhitungan *hidden layer.*

Tabel 4.9 Matriks Keluaran *Hidden Layer*

H_{init}	1	2
1	5,280182	5,782626
2	7,850341	7,59408
3	7,835376	8,512116
4	4,111579	5,227085
5	8,671808	8,088946
6	5,679355	6,569778
7	6,469878	7,222899
8	6,825344	7,370412
9	7,74749	8,006607
10	7,095675	6,958305
11	7,677121	7,443651
12	7,59667	7,830241
13	6,984771	7,461405
14	7,700127	7,331867
15	8,786432	8,01166
16	7,574983	7,844746
17	6,847273	7,045258
18	7,294563	7,14054
19	7,295475	6,991803
20	7,426908	7,137586

Kemudian hasil dari proses *hidden layer* akan dilakukan perhitungan dengan fungsi aktivasi. Contoh proses perhitungan dari *hidden layer* dengan fungsi aktivasi.

$$H(x)_{1,1} = \frac{1}{1+e^{-x}} = \frac{1}{1+e^{-(5,280182)}} = 0,994934$$

Tabel 4.10 adalah hasil dari proses perhitungan *hidden layer* menggunakan fungsi aktivasi.

Tabel 4.10 Matriks Keluaran *Hidden Layer* Dengan Fungsi Aktivasi

H(x)	1	2
1	0,994934	0,996929
2	0,999611	0,999497

Tabel 4.10 Matriks Keluaran *Hidden Layer* Dengan Fungsi Aktivasi (lanjutan)

H(x)	1	2
3	0,999605	0,999799
4	0,983882	0,99466
5	0,999829	0,999693
6	0,996596	0,9986
7	0,998453	0,999271
8	0,998915	0,999371
9	0,999568	0,999667
10	0,999172	0,99905
11	0,999537	0,999415
12	0,999498	0,999603
13	0,999075	0,999425
14	0,999547	0,999346
15	0,999847	0,999669
16	0,999487	0,999608
17	0,998939	0,999129
18	0,999321	0,999208
19	0,999322	0,999081
20	0,999405	0,999206

Dari perhitungan keseluruhan *hidden layer* menggunakan fungsi aktivasi nantinya akan dipergunakan dalam melakukan perhitungan untuk bobot keluaran. Selanjutnya melakukan proses *transpose* untuk *hidden layer* dengan fungsi aktivasi. Tabel 4.11 adalah hasil proses *transpose* dari matriks *hidden layer* menggunakan fungsi aktivasi.

Tabel 4.11 *Transpose* Matriks Keluaran Hidden Layer Dengan Fungsi Aktivasi

HT	**1**	**2**	**3**	**4**	**...**	**16**	**17**	**18**	**19**	**20**
1	0,995	1,000	1,000	0,984	...	0,999	0,999	0,999	0,999	0,999
2	0,997	0,999	1,000	0,995	...	1,000	0,999	0,999	0,999	0,999

Kemudian melakukan perkalian antara matriks Htranspose dengan matriks H. Contoh proses perhitungan dari matriks Htranspose dengan matriks H.

$$(H^T H) \quad = (H^T * H)$$

$= ((0{,}995 * 0{,}994934)+(1{,}000*0{,}999611)+(1{,}000*0{,}999605)+$
$(0{,}984*0{,}983882)+(1{,}000*0{,}999829)+(0{,}997*0{,}996596)+(0{,}998*0{,}998453)+(0{,}999*0{,}998915)+(1{,}000*0{,}999568)+(0{,}999*0{,}999172)+(1{,}000*0{,}999537)+(0{,}999*0{,}999498)+(0{,}999*0{,}999075)+(1{,}000*0{,}999547)+(1{,}000*0{,}999847)+(0{,}999*0{,}999487)+(0{,}999*0{,}998939)+(0{,}999*0{,}999321)+(0{,}999*0{,}999322)+(0{,}999*0{,}999405)) = 19{,}92939$

Tabel 4.12 adalah hasil proses perkalian matriks H*transpose* dengan matriks H.

Tabel 4.12 Hasil Perkalian Matrik *Transpose* Dengan Matriks Keluaran *Hidden Layer*

$H^T * H$	1	2
1	19,92939	19,94488
2	19,94488	19,9605

Langkah selanjutnya yaitu melakukan perhitungan *invers* hasil perkalian Htranspose dengan matriks H. Untuk perhitungan nilai *invers* akan menggunakan operasi baris elementer. Operasi dari baris elementer sendiri adalah solusi untuk penyelesaian masalah seperti halnya menentukan nilai dari suatu *invers* melalui cara melakukan perubahan pada matriks yang akan dilakuan *invers* menjadi suatu matriks indentitas. Cara untuk melakukan perhitungan dengan operasi baris elementer yaitu menyandingkan nilai dari matriks yang nantinya akan dilakukan *invers* dengan nilai matriks identitas. Setelah itu melakukan perubahan pada matriks yang akan dilakukan *invers* menjadi sebuah matriks identitas. Sedangkan matriks identitas akan berubah menjadi matriks yang baru. Nilai dari matriks baru yang didapatkan dari matriks identitas tadi merupakan hasil dari proses *invers*. Contoh perhitungan dengan operasi baris elementer.

1. Bentuk matriks identitas (*I*)

$$[H \mid I] = \begin{bmatrix} 19{,}92939 & 19{,}94488 \\ 19{,}94488 & 19{,}9605 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

2. $\mathrm{R2} \div 19{,}94488 \rightarrow \mathrm{R2}$

$$[H \mid I] = \begin{bmatrix} 19{,}92939 & 19{,}94488 \\ 1 & 1{,}00078 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 0{,}05014 \end{bmatrix}$$

3. $\mathrm{R2} - (1/19{,}92939 \text{ x } \mathrm{R1}) \rightarrow \mathrm{R2}$

$$[H \mid I] = \begin{bmatrix} 19{,}92939 & 19{,}94488 \\ 0 & 0{,}00000591432 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -0{,}05017715043 & 0{,}05014 \end{bmatrix}$$

4. $\mathrm{R1} \div 19{,}94488 \rightarrow \mathrm{R1}$

$$[H \mid I] = \begin{bmatrix} 0{,}99922335958 & 1 \\ 0 & 0{,}00000591432 \end{bmatrix} \begin{bmatrix} 0{,}05013818083 & 0 \\ -0{,}05017715043 & 0{,}05014 \end{bmatrix}$$

5. $\mathrm{R1} - (1/0{,}00000591432 \text{ x } \mathrm{R2}) \rightarrow \mathrm{R1}$

$$[H \mid I] = \begin{bmatrix} 0{,}999223359 & 0 \\ 0 & 0{,}0000059143 \end{bmatrix} \begin{bmatrix} 8820{,}785 & -8813{,}8795 \\ -0{,}05017715 & 0{,}05014 \end{bmatrix}$$

6. $R1 \div 0{,}99922335958 \rightarrow R1$

$$[H \mid I] = \begin{bmatrix} 1 & 0 \\ 0 & 0{,}00000591432 \end{bmatrix} \begin{bmatrix} 8827{,}640903 & -8820{,}730036 \\ 0{,}052168619 & 0{,}05212813 \end{bmatrix}$$

7. $R2 \div 0{,}00000591432 \rightarrow R2$

$$[H \mid I] = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 8827{,}640903 & -8820{,}730036 \\ -8820{,}73 & 8813{,}884 \end{bmatrix}$$

Tabel 4.13 merupakan hasil dari *invers* matriks H^T*H.

Tabel 4.13 Invers Matriks

$(H^T * H)^{-1}$	1	2
1	8827,641	-8820,73
2	-8820,73	8813,884

Melakukan proses perhitungan untuk mencari nilai dari *moore-penrose (H+)*. Contoh proses perhitungan matriks *moore-penrose.*

$$H^+ = (H^T H)^{-1} H^T$$

$$H^+ = ((\ 8827{,}641*0{,}995)+(\ -8820{,}73*0{,}997)) = -10{,}72$$

Tabel 4.14 adalah hasil proses perhitungan *moore-penrose.*

Tabel 4.14 Matriks *Moore-Penrose Generalized Invers*

H+	1	2	3	4	...	17	18	19	20
1	-10,72	7,91	5,19	-88,27	...	5,22	7,90	9,02	8,66
2	10,76	-7,85	-5,14	88,25	...	-5,17	-7,84	-8,97	-8,60

Langkah selanjutnya yaitu proses menghitung *output weight.* Dalam perhitungan ini akan melibatkan matriks targen (*T*) yang sudah di *transpose* serta matriks *moore-penrose* (H^+). Dari proses ini akan mendapatkan nilai *output weight.* Tabel 4.15 merupakan matriks target (*T*).

Tabel 4.15 Matriks Target (T)

T	1	2	3	4	...	17	18	19	20
1	1	1	1	1	...	-1	-1	-1	-1
2	-1	-1	-1	-1	...	1	1	1	1

Setelah membuat matriks T, maka matriks tersebut selanjutnya di *transpose.* Tabel 4.16 merupakan matriks T yang sudah di *transpose*.

Tabel 4.16 Matriks T *Transpose*

T	1	2
1	1	-1
2	1	-1
3	1	-1
4	1	-1
5	1	-1
6	1	-1
7	1	-1
8	1	-1
9	1	-1
10	-1	1
11	-1	1
12	-1	1
13	-1	1
14	-1	1
15	-1	1
16	-1	1
17	-1	1
18	-1	1
19	-1	1
20	-1	1

Dibawah ini contoh perhitungan dari *output weight* (*β*).

$$\beta = H^{+} T$$

β = ((-10,72*1)+(7,91*1)+(5,19*1)+(-88,27*1)+(8,10*1)+(-10,79*1)+(-0,32*1)+(2,88*1)+(6,03*1)+(7,98*-1)+(7,98*-1)+(5,98*-1)+(3,81*-1)+(8,68*-1)+(8,48*-1)+(5,83*-1)+(5,22*-1)+(7,90*-1)+(9,02*-1)+(8,66*-1)) = -159,528

Tabel 4.17 merupakan nNilai keseluruhan dari β.

Tabel 4.17 Nilai *Output Weight*

β	1	2
1	-159,528	159,5279
2	159,3026	-159,303

Langkah berikutnya yaitu menghitung Y prediksi atau keluaran *output layer* untuk proses *training*. Berikut ini contoh perhitungan dari Y prediksi.

$$Y = H\ \beta$$

$$Y = (0{,}994934 * -159{,}528) + (0{,}996929 * 159{,}3026) = 0{,}09361$$

Tabel 4.18 merupakan nilai dari Y prediksi.

Tabel 4.18 Matriks Y Prediksi Proses *Training*

Y	1	2
1	0,09361	-0,09361
2	-0,2433	0,243295
3	-0,19422	0,194219
4	1,495224	-1,49522
5	-0,24682	0,246817
6	0,094737	-0,09474
7	-0,09463	0,094633
8	-0,15246	0,152461
9	-0,20948	0,209482
10	-0,24449	0,244488
11	-0,24455	0,244555
12	-0,20851	0,20851
13	-0,16923	0,169226
14	-0,25724	0,257245
15	-0,2537	0,253702
16	-0,20584	0,205845
17	-0,19469	0,194692
18	-0,24311	0,243108
19	-0,26341	0,263414
20	-0,2569	0,256896

Proses selanjutnya yaitu menentukan kelas hasil klasifikasi dari proses *training* apakah sesuai dengan kelas aslinya atau tidak. Tabel 4.19 merupakan hasil dari klasifikasi kelas untuk proses training.

Tabel 4.19 Klasifikasi Kelas Untuk Proses *Training*

No	Y = 1	Y = 2	Kelas asli	Kelas klasifikasi
1	0,093609618	-0,093609618	1	1
2	-0,24329534	0,24329534	1	2
3	-0,194219014	0,194219014	1	2
4	1,495223803	-1,495223803	1	1
5	-0,246816876	0,246816876	1	2
6	0,094737193	-0,094737193	1	1
7	-0,094633366	0,094633366	1	2
8	-0,15246148	0,15246148	1	2
9	-0,20948216	0,20948216	1	2
10	-0,244488053	0,244488053	2	2
11	-0,244554528	0,244554528	2	2
12	-0,208510358	0,208510358	2	2
13	-0,169226013	0,169226013	2	2
14	-0,257244983	0,257244983	2	2
15	-0,253702234	0,253702234	2	2
16	-0,205844753	0,205844753	2	2
17	-0,194692138	0,194692138	2	2
18	-0,24310827	0,24310827	2	2
19	-0,26341374	0,26341374	2	2
20	-0,256895708	0,256895708	2	2

4.3.3 Perhitungan Manual Proses *Testing*

Dalam penelitian ini menggunakan fungsi aktivasi sigmoid, data yang digunakan dalam perhitungan manual proses *testing* berjumlah 5 data dan juga dalam penelitian ini menggunakan jumlah *neuron hidden layer* sebanyak 2. Data ini didapatkan dari presentase antara data untuk *training* serta data untuk *testing* yang mana 80% data untuk *training* serta 20% data untuk *testing* dari jumlah keseluruhan data dalam perhitungan manual sebanyak 25 data. Tahapan proses *testing* dengan algoritma *Extreme Learning Machine*:

Tabel 4.19 adalah nilai normalisasi dari data *testing*.

Tabel 4.20 Normalisasi Data *Testing*

no	X1	X2	X3	X4	X5	...	X20	X21	X22	X23	X24
21	0,821918	0,333333	1	0	0	...	1	1	0	1	1
22	0,931507	0	0,75	0	0	...	1	1	0	1	1
23	0,369863	0,666667	1	0	0	...	1	1	0	1	1
24	0,328767	0,666667	0,75	0	0	...	1	1	0	1	1
25	0,890411	0,333333	0,75	0	0	...	1	1	0	1	1

Selanjutnya menetukan nilai bias untuk proes *testing*. Tabel 4.20 merupakan nilai matriks bias.

Tabel 4.21 Nilai Matriks Bias

No	1	2
21	0,408168	0,899462
22	0,351978	0,871721
23	0,810902	0,644251
24	0,278122	0,414571
25	0,994694	0,303787

Langkah pertama dalam perhitungan *testing* yaitu menghitung nilai dari matriks *hidden layer* (*Hinit)*. Contoh proses perhitungan dari *hidden layer.*

$$H_{init} = (X_{test} * W^T) + b$$

H_{init} = ((0,821918*0,17)+(0,333333*0,22)+(1*0,48)+(0* 0,67)+(0* 0,68)+(0* 0,63)+(0* 0,42)+(1* 0,15)+(1* 0,40)+(0,091691* 0,86)+(0,007* 0,92)+(0,053*0,57)+(0,806* 1,00)+(0,025*0,50)+(0,621* 0,52)+(0,613* 0,08)+(0,513*0,99)+(0,571* 0,93)+(1,000* 0,93)+(1* 0,18)+(1* 0,50)+(0*0,35)+(1*1,00)+(1*0,89) +0,408168)= 7,475568

Tabel 4.21 adalah hasil nilai perhitungan *hidden layer*.

Tabel 4.22 Matriks Keluaran *Hidden Layer*

H_{init}	1	2
21	7,475568	7,659649
22	7,273672	7,847954
23	7,95839	7,253779
24	7,603385	7,278811
25	8,297846	7,14157

Kemudian menghitung nilai *hidden layer* dengan menggunakan fungsi aktivasi. Berikut contoh perhitungannya:

$$H(x)_{1,1} = \frac{1}{1+e^{-x}} = \frac{1}{1+e^{-(7,475568)}} = 0,999434$$

Tabel 4.22 adalah nilai dari proses perhitungan *hidden layer* menggunakan fungsi aktivasi.

Tabel 4.23 Matriks Keluaran *Hidden Layer* Dengan Fungsi Aktivasi

H(x)	1	2
21	0,999434	0,999529
22	0,999307	0,99961
23	0,99965	0,999293
24	0,999501	0,99931
25	0,999751	0,999209

Selanjutnya menghitung nilai *output layer* sebagai hasil klasifikasi. Dalam menghitung nilai *output layer* akan menggunakan nilai *output* bobot proses *training* pada perhitungan sebelumnya. Berikut proses perhitungan untuk keluaran *output layer*.

$$y = H\,\beta$$

$$y = ((0,999434* -159,528)+(0,999529*159,3026)) = -0,20998$$

Tabel 4.23 adalah hasil perhitungan dari *output layer.*

Tabel 4.24 Keluaran *Output Layer*

y	1	2
21	-0,20998	0,209978
22	-0,1769	0,176896
23	-0,28213	0,282126
24	-0,25559	0,255588
25	-0,31154	0,311539

Proses selanjutnya yaitu menentukan kelas hasil klasifikasi dari proses *testing* apakah sesuai dengan kelas aslinya atau tidak. Tabel 4.25 merupakan hasil dari klasifikasi kelas untuk proses training.

Tabel 4.25 Klasifikasi Kelas Untuk Proses *Testing*

No	Y = 1	Y = 2	Kelas asli	Kelas klasifikasi
21	-0,20998	0,209978	2	2
22	-0,1769	0,176896	2	2

Tabel 4.2526 Klasifikasi Kelas Untuk Proses *Testing* (lanjutan)

No	Y = 1	Y = 2	Kelas asli	Kelas klasifikasi
23	-0,28213	0,282126	2	2
24	-0,25559	0,255588	2	2
25	-0,31154	0,311539	2	2

4.3.4 Perhitungan Nilai Akurasi

Dalam hal ini akan melakukan proses perhitungan evaluasi dengan mendapatkan nilai akurasi. Berikut perhitungan keluaran dari nilai akurasi.

$$Akurasi = \frac{DT}{N} * 100\%$$

$$Akurasi = \frac{5}{5} * 100\%$$

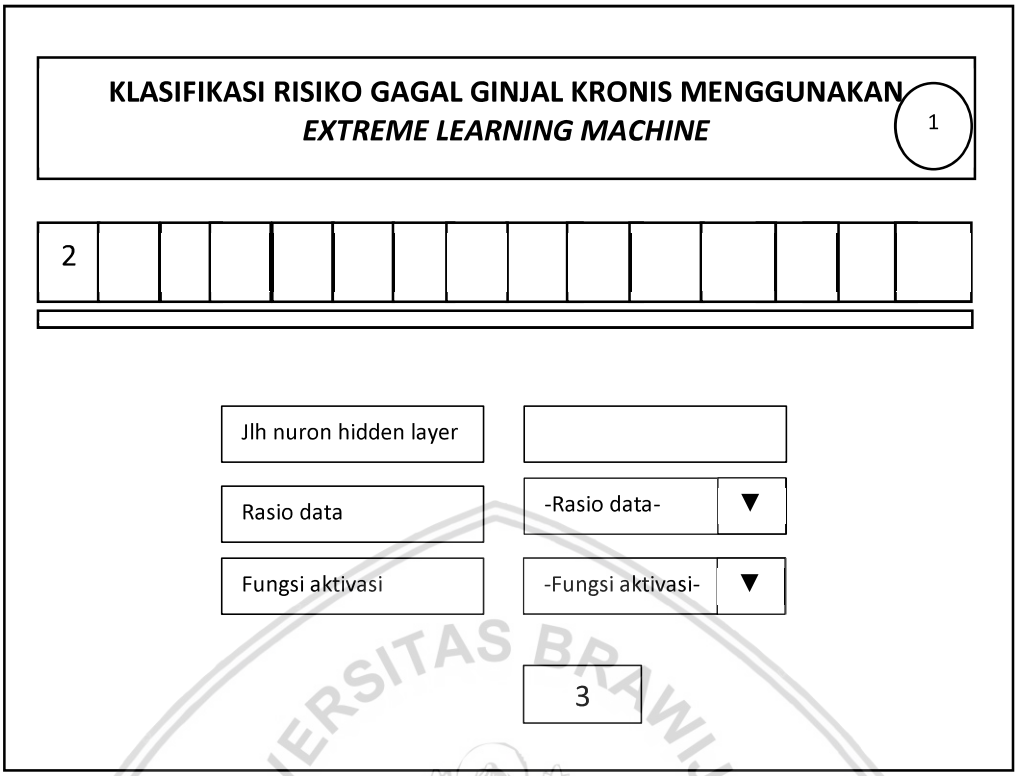
$$Akurasi = 100\ \%$$

4.4 Perancangan Antarmuka

Dalam perancangan antarmuka ini bertujuan untuk menggambarkan implementasi dari klasifikasi gagal ginjal kronis. Antarmuka dalam penelitian ini terdiri dari halaman inisialisasi parameter, data set, *normalisasi,* pembagian data, h*idden layer*, *hidden layer* dengan fungsi aktivasi, *htranspose, MoorePenrose, output weight, output layer,* data uji*, Hidden layer* data uji*, hidden layer* dengan fungsi aktivasi data uji*, output layer* data uji*,* hasil klasifikasi.

4.4.1 Perancangan Halaman Inisialisasi Parameter

Halaman ini merupakan tampilan pertama dari program aplikasi yang akan dibuat. Gambar 4.21 merupakan rancangan antar muka halaman inisialisasi parameter pada *program* aplikasi.



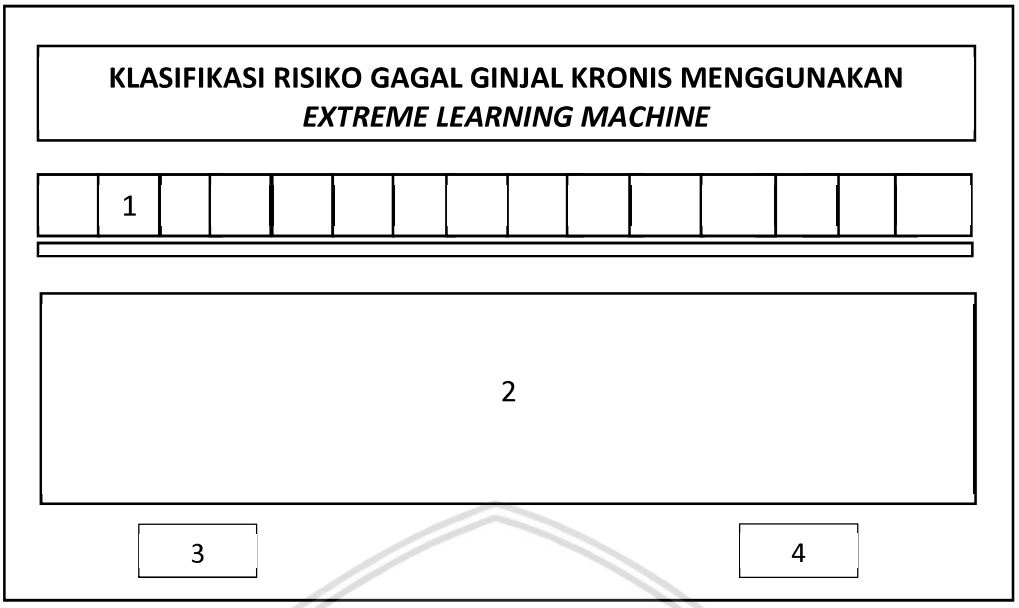
Gambar 4.21 Rancangan Halaman Inisialisasi Parameter

Keterangan dari Gambar 4.21 yang mana sebagai rancangan halaman inisialisasi parameter dari *program*:

1. Header judul dari program aplikasi.
2. Tab fungsi untuk menginputkan inisialisasi parameter.
3. *Button* untuk memproses nilai inisialisasi parameter.

4.4.2 Perancangan Halaman *Dataset*

Halaman ini merupakan tampilan kedua dari program aplikasi yang akan dibuat. Gambar 4.22 merupakan rancangan antar muka halaman *dataset* pada program aplikasi.



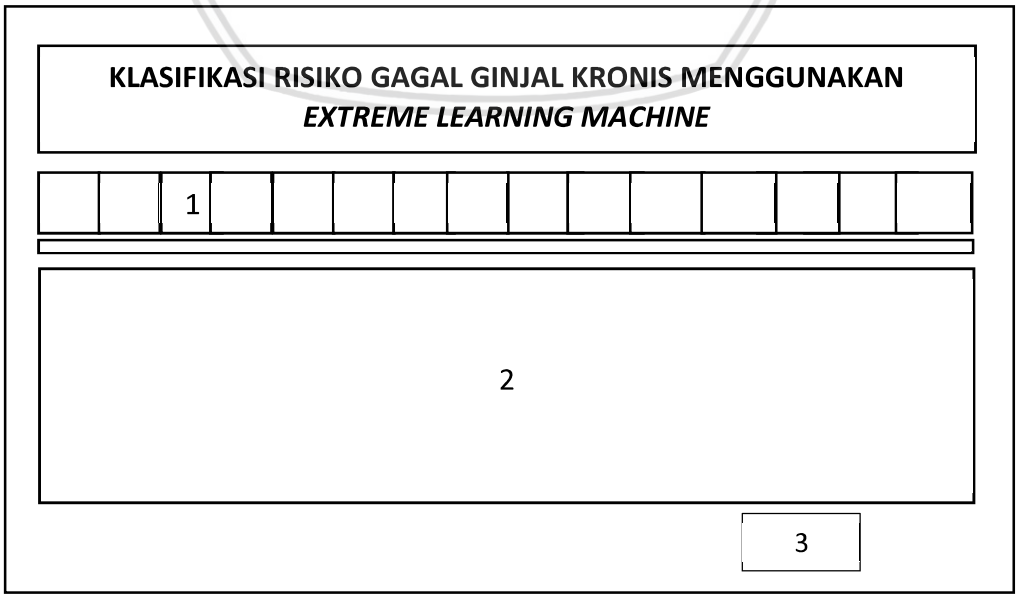
Gambar 4.22 Rancangan Halaman *Dataset*

Keterangan dari Gambar 4.22 yang mana sebagai rancangan halaman *dataset* dari *program*:

1. Tab fungsi untuk menginputkan nilai *dataset*.
2. Tabel untuk menampilkan nilai *dataset.*
3. *Button* untuk memanggil nilai *dataset.*
4. *Button* untuk proses *normalisasi.*

4.4.3 Perancangan Halaman *Normalisasi*

Halaman ini merupakan tampilan ketiga dari program aplikasi yang akan dibuat. Gambar 4.23 merupakan rancangan antar muka halaman *normalisasi* pada program aplikasi.



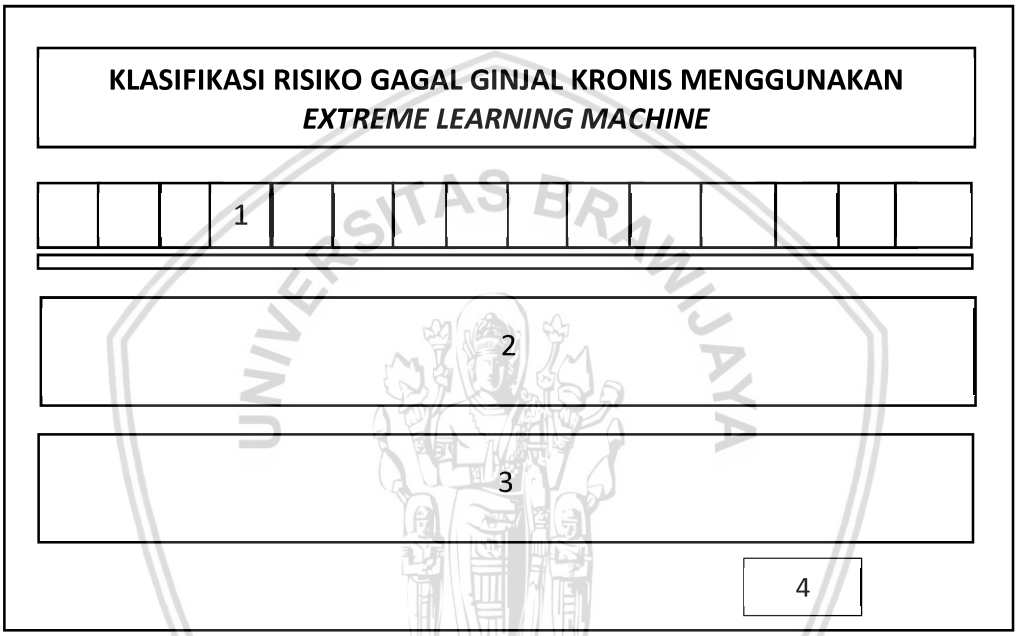
Gambar 4.23 Rancangan Halaman *Normalisasi*

Keterangan dari Gambar 4.23 yang mana sebagai rancangan halaman *normalisasi* dari *program*:

1. Tab fungsi untuk nilai *normalisasi*.
2. Tabel untuk menampilkan nilai *normalisasi.*
3. *Button* untuk proses pembagian data.

4.4.4 Perancangan Halaman Pembagian Data

Halaman ini merupakan tampilan keempat dari program aplikasi yang akan dibuat. Gambar 4.24 merupakan rancangan antar muka halaman pembagian data pada program aplikasi.



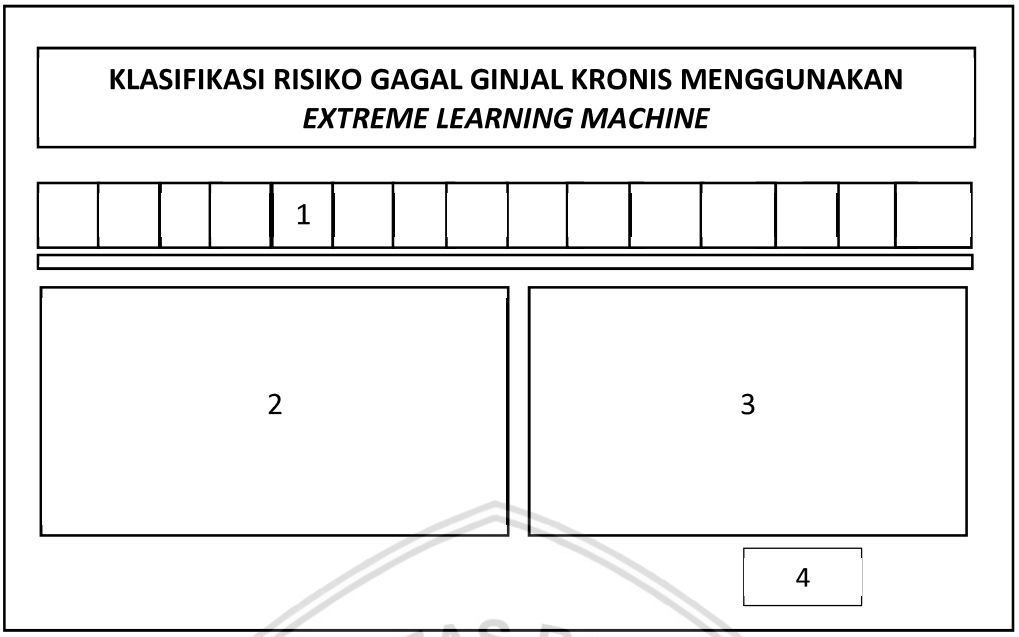
Gambar 4.24 Rancangan Halaman Pembagian Data

Keterangan dari Gambar 4.24 yang mana sebagai rancangan halaman pembagian data dari *program*:

1. Tab fungsi untuk nilai pembagian data.
2. Tabel untuk menampilkan nilai data latih.
3. Tabel untuk menampilkan nilai data uji.
4. *Button* untuk proses *hidden layer* data latih.

4.4.5 Perancangan Halaman *Hidden Layer* Data Latih

Halaman ini merupakan tampilan kelima dari program aplikasi yang akan dibuat. Gambar 4.25 merupakan rancangan antar muka halaman *hidden layer* data latih pada program aplikasi.



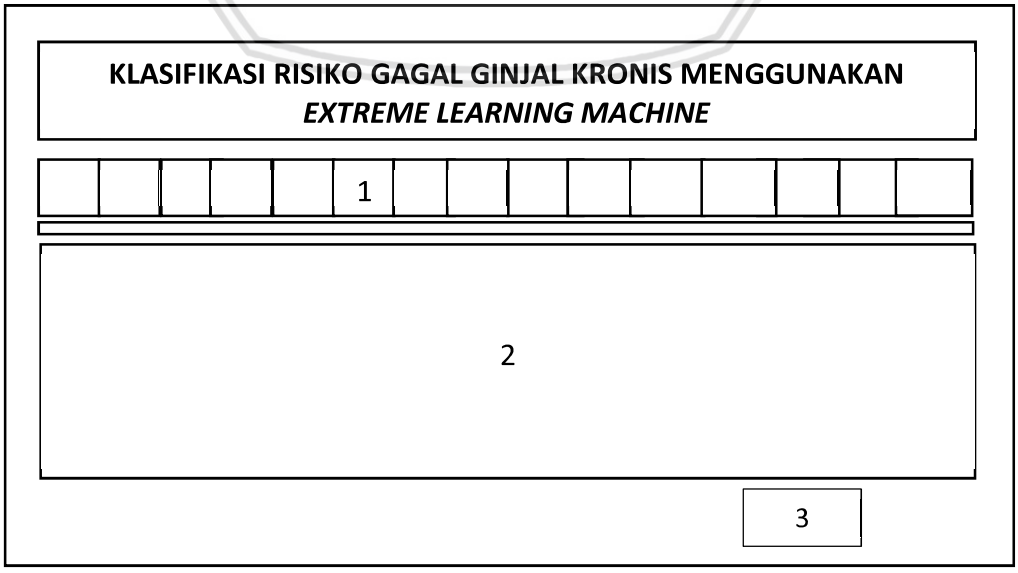
Gambar 4.25 Rancangan Halaman *Hidden Layer* Data Latih

Keterangan dari Gambar 4.25 yang mana sebagai rancangan halaman *hidden layer* data latih dari *program*:

1. Tab fungsi untuk nilai *hidden layer*.
2. Tabel untuk menampilkan nilai *weight*.
3. Tabel untuk menampilkan nilai *hidden layer*.
4. *Button* untuk proses *hidden layer* dengan fungsi aktivasi.

4.4.6 Perancangan Halaman *Hidden Layer* dengan Fungsi Aktivasi

Halaman ini merupakan tampilan keenam dari program aplikasi yang akan dibuat. Gambar 4.26 merupakan rancangan antar muka halaman *hidden layer* dengan fungsi aktivasi pada program aplikasi.



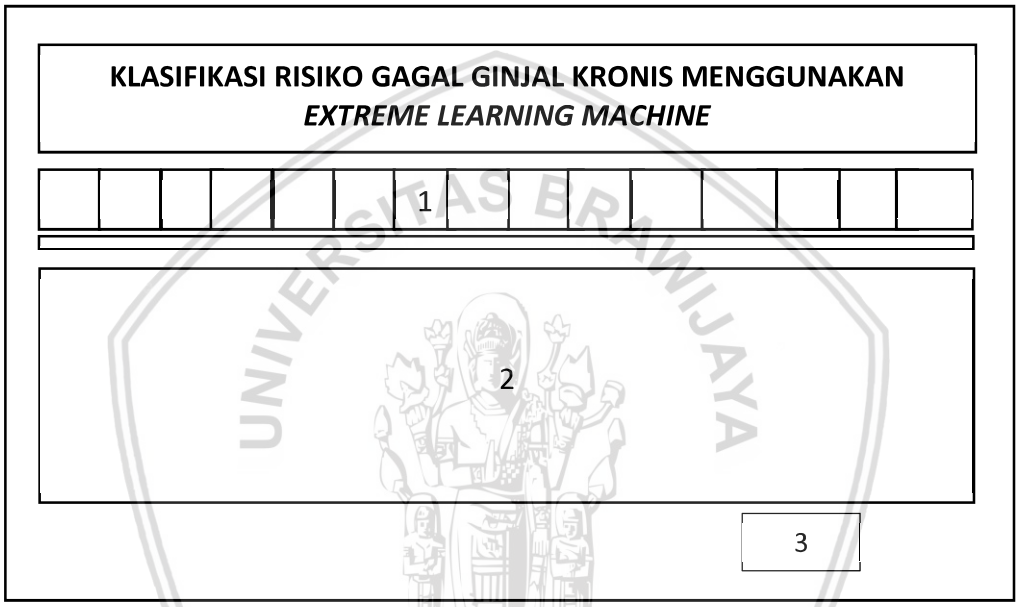
Gambar 4.26 Rancangan Halaman Hidden Layer Dengan Fungsi Aktivasi

Keterangan dari Gambar 4.26 yang mana sebagai rancangan halaman *hidden layer* dengan fungsi aktivasi dari *program*:

1. Tab fungsi untuk nilai *hidden layer* dengan fungsi aktivasi.
2. Tabel untuk menampilkan *hidden layer* dengan fungsi aktivasi.
3. *Button* untuk proses *HTranspose.*

4.4.7 Perancangan Halaman *Htranspose*

Halaman ini merupakan tampilan ketujuh dari program aplikasi yang akan dibuat. Gambar 4.27 merupakan rancangan antar muka halaman *Htranspose* pada program aplikasi.



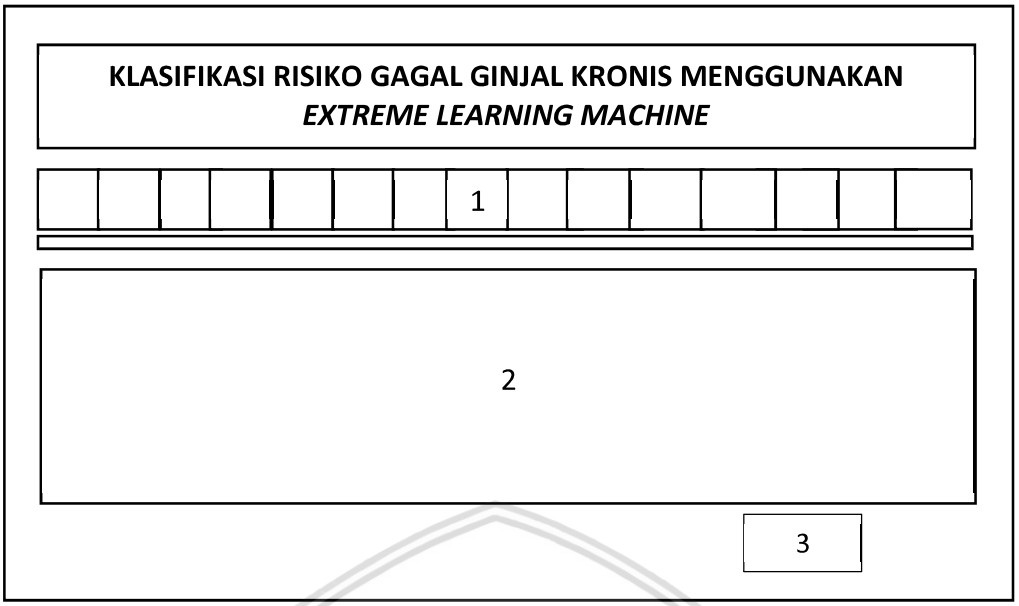
Gambar 4.27 Rancangan Halaman *Htranspose*

Keterangan dari Gambar 4.27 yang mana sebagai rancangan halaman *Htranspose* dari *program*:

1. Tab fungsi untuk nilai *Htranspose*.
2. Tabel untuk menampilkan nilai *Htranspose.*
3. *Button* untuk proses *MoorePenrose.*

4.4.8 Perancangan Halaman *MoorePenrose*

Halaman ini merupakan tampilan kedelapan dari program aplikasi yang akan dibuat. Gambar 4.28 merupakan rancangan antar muka halaman *MoorePenrose* pada program aplikasi.



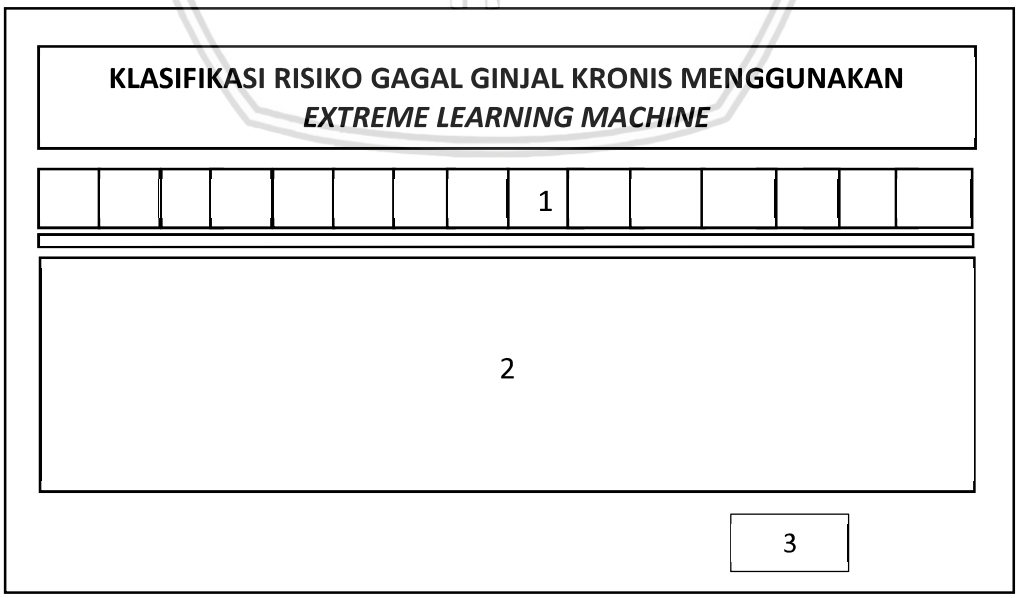
Gambar 4.28 Rancangan Halaman *MoorePenrose*

Keterangan dari Gambar 4.28 yang mana sebagai rancangan halaman *MoorePenrose* dari *program*:

1. Tab fungsi untuk nilai *MoorePenrose*.
2. Tabel untuk menampilkan nilai *MoorePenrose.*
3. *Button* untuk proses *output weight.*

4.4.9 Perancangan Halaman *Output Weight*

Halaman ini merupakan tampilan kesembilan dari program aplikasi yang akan dibuat. Gambar 4.29 merupakan rancangan antar muka halaman *output weight* pada program aplikasi.



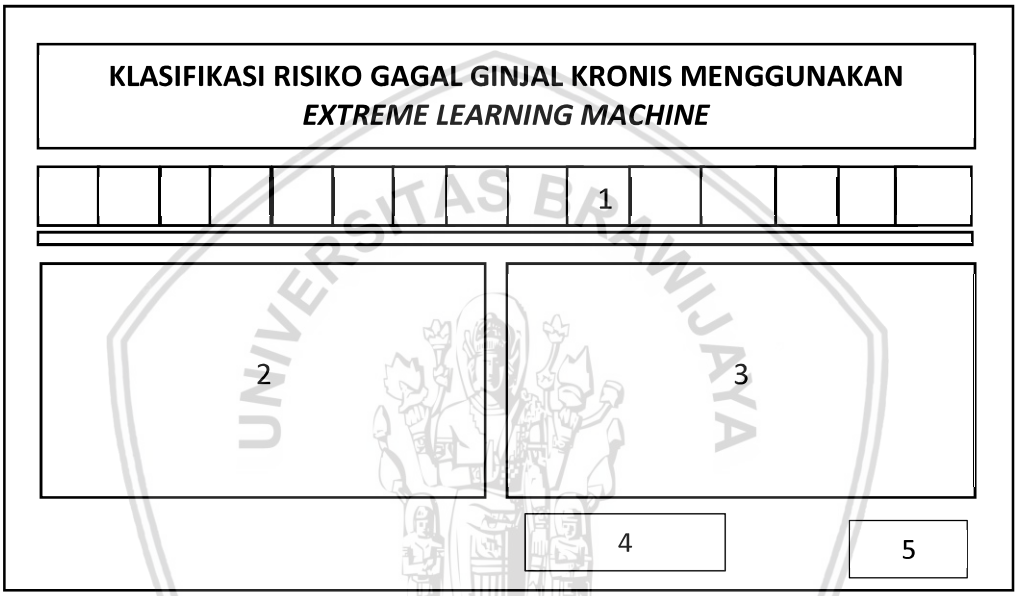
Gambar 4.29 Rancangan Halaman *Output Weight*

Keterangan dari Gambar 4.29 yang mana sebagai rancangan halaman *output weight* dari *program*:

1. Tab fungsi untuk nilai *output weight*.
2. Tabel untuk menampilkan nilai o*utput weight.*
3. *Button* untuk proses *output layer.*

4.4.10 Perancangan Halaman *Output Layer*

Halaman ini merupakan tampilan kesepuluh dari program aplikasi yang akan dibuat. Gambar 4.30 merupakan rancangan antar muka halaman *output layer* pada program aplikasi.



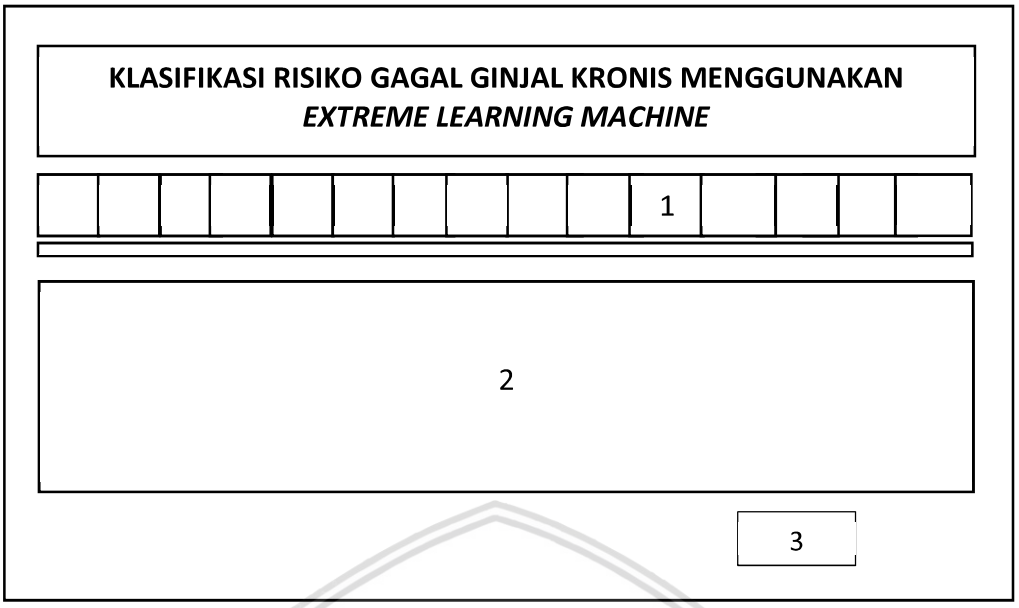
Gambar 4.30 Rancangan Halaman *Output Layer*

Keterangan dari Gambar 4.30 yang mana sebagai rancangan halaman *output layer* dari *program*:

1. Tab fungsi untuk nilai *output layer*.
2. Tabel untuk menampilkan nilai o*utput layer.*
3. Tabel untuk menampilkan nilai klasifikasi data latih.
4. Label untuk menampilkan nilai akurasi data latih.
5. *Button* untuk proses menampilkan data uji.

4.4.11 Perancangan Halaman Data Uji

Halaman ini merupakan tampilan kesebelas dari program aplikasi yang akan dibuat. Gambar 4.31 merupakan rancangan antar muka halaman data uji pada program aplikasi.



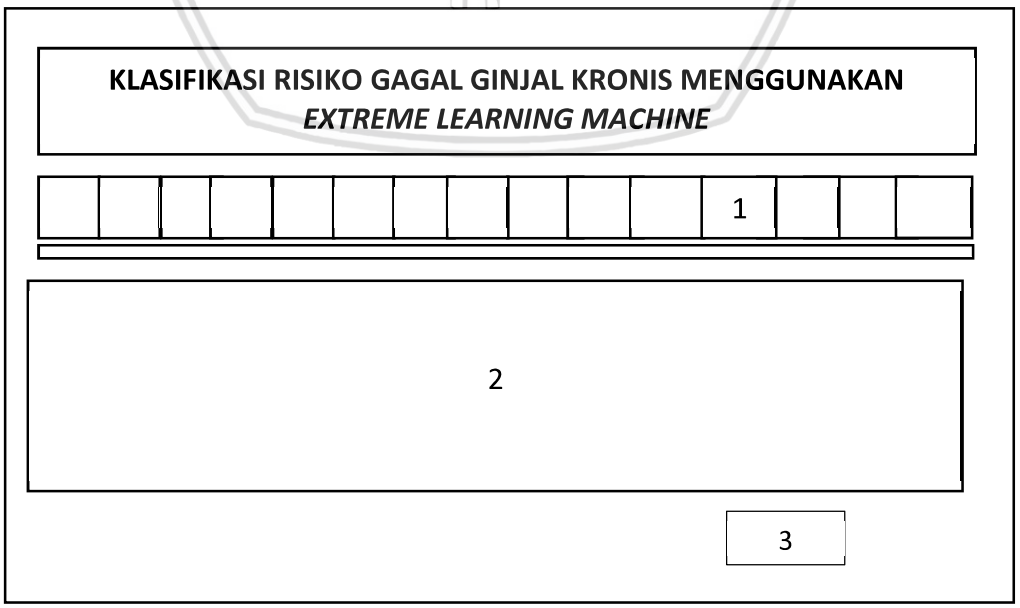
Gambar 4.31 Rancangan Halaman Data Uji

Keterangan dari Gambar 4.31 yang mana sebagai rancangan halaman data uji dari *program*:

1. Tab fungsi untuk nilai data latih.
2. Tabel untuk menampilkan nilai data latih.
3. *Button* untuk proses *hidden layer* data uji.

4.4.12 Perancangan Halaman *Hidden Layer* Data Uji

Halaman ini merupakan tampilan keduabelas dari program aplikasi yang akan dibuat. Gambar 4.32 merupakan rancangan antar muka halaman *hidden layer* data uji pada program aplikasi.



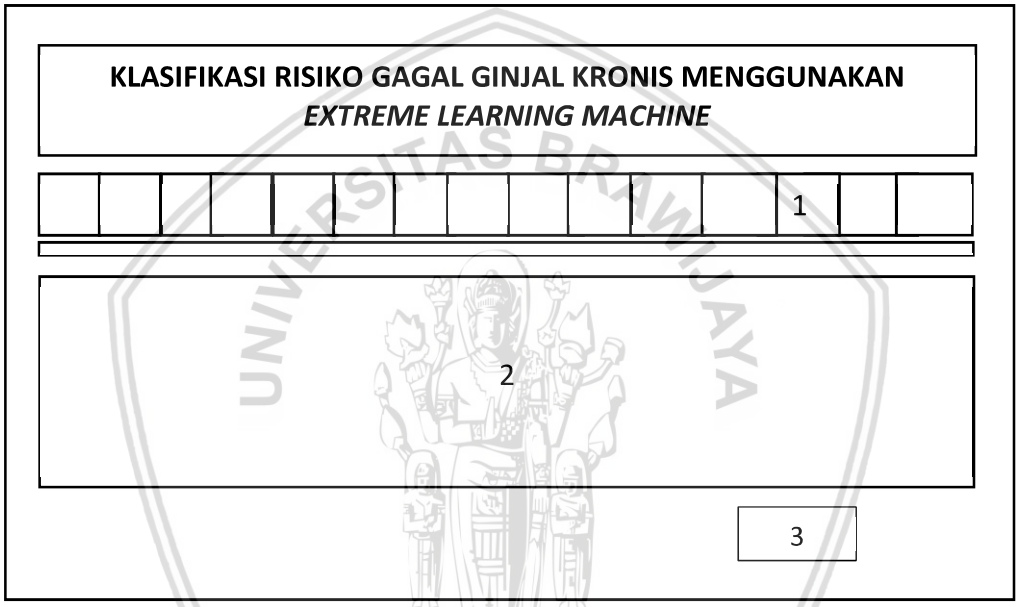
Gambar 4.32 Rancangan Halaman *Hidden Layer* Data Uji

Keterangan dari Gambar 4.32 yang mana sebagai rancangan halaman *hidden layer* data uji dari *program*:

1. Tab fungsi untuk nilai *hidden layer* data uji.
2. Tabel untuk menampilkan nilai *hidden layer* data uji.
3. *Button* untuk proses *hidden layer* dengan fungsi aktivasi data uji.

4.4.13 Perancangan Halaman *Hidden Layer* dengan Fungsi Aktivasi Data Uji

Halaman ini merupakan tampilan ketigabelas dari program aplikasi yang akan dibuat. Gambar 4.33 merupakan rancangan antar muka halaman *hidden layer* dengan fungsi aktivasi data uji pada program aplikasi.



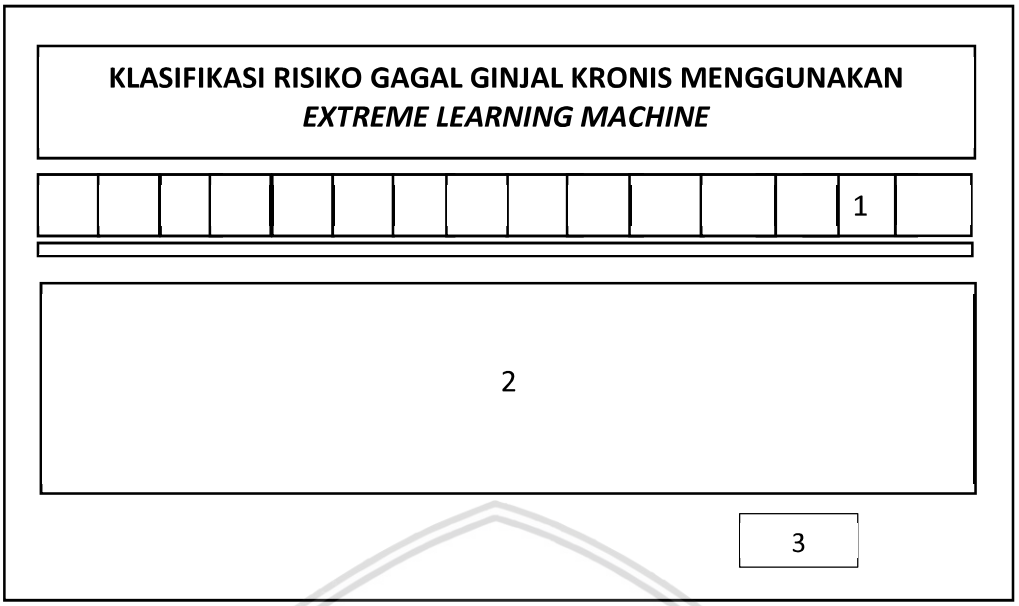
Gambar 4.33 Rancangan Halaman *Hidden Layer* dengan Fungsi Aktvasi Data Uji

Keterangan dari Gambar 4.33 yang mana sebagai rancangan halaman *hidden layer* dengan fungsi aktivasi data uji dari *program*:

1. Tab fungsi untuk nilai *hidden layer* dengan fungsi aktvasi data uji.
2. Tabel untuk menampilkan nilai *hidden layer* dengan fungsi aktivasi data uji.
3. *Button* untuk proses *output layer* data uji.

4.4.14 Perancangan Halaman *Output Layer* Data Uji

Halaman ini merupakan tampilan keempatbelas dari program aplikasi yang akan dibuat. Gambar 4.34 merupakan rancangan antar muka halaman *output layer* data uji pada program aplikasi.



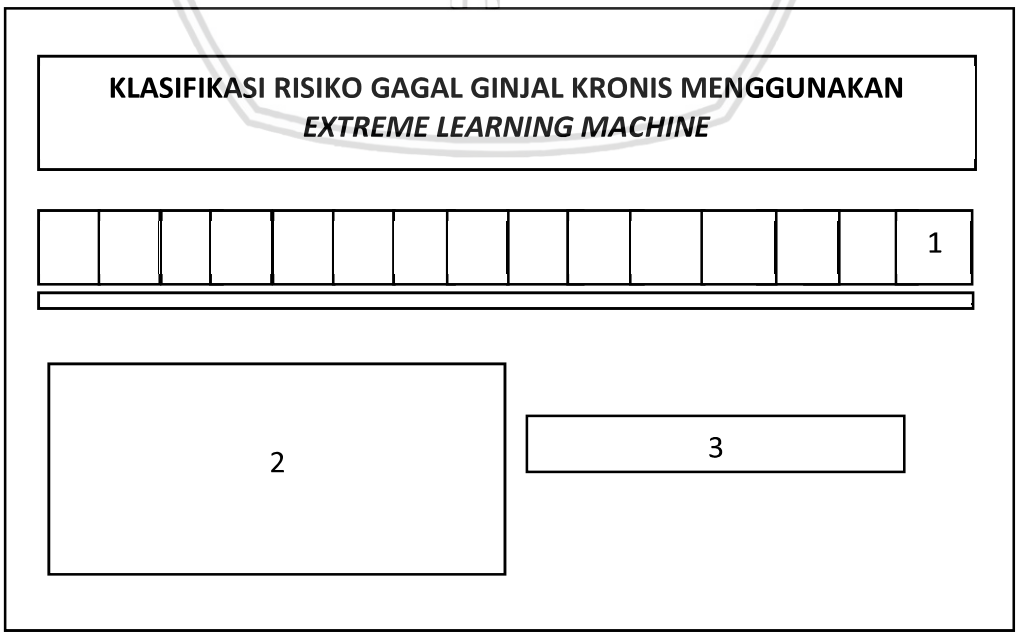
Gambar 4.34 Rancangan Halaman *Output Layer* Data Uji

Keterangan dari Gambar 4.34 yang mana sebagai rancangan *output layer* data uji dari *program*:

1. Tab fungsi untuk nilai *output layer* data uji.
2. Tabel untuk menampilkan nilai *output layer* data uji.
3. *Button* untuk proses klasifikasi data uji.

4.4.15 Perancangan Halaman Hasil Klasifikasi

Halaman ini merupakan tampilan kelimabelas dari program aplikasi yang akan dibuat. Gambar 4.35 merupakan rancangan antar muka halaman hasil klasifikasi data uji pada program aplikasi.



Gambar 4.35 Rancangan Halaman Hasil Klasifikasi Data Uji

Keterangan dari Gambar 4.35 yang mana sebagai rancangan hasil klasifikasi data uji dari *program*:

1. Tab fungsi untuk nilai hasil klasifikasi.
2. Tabel untuk menampilkan nilai hasil klasifikasi.
3. Label untuk nilai akurasi hasil klasifikasi data uji.

4.5 Perancangan Pengujian

Pengujian yang dilakukan adalah pengujian yang ada kaitannya dengan *Extreme Learning Machine* yang digunakan dalam klasifikasi gagal ginjla kronis. Pengujian yang akan dipergunakan yaitu:

1. Pengujian untuk perbandingan rasio data latih serta data uji.
2. Pengujian untuk pengaruh jumlah *neuron hidden layer*.
3. Pengujian untuk perbandingan fungsi aktivasi.

4.5.1 Pengujian untuk Rasio Data Latih dan Data Uji

Pengujian rasio jumlah data latih serta data uji bertujuan mengetahui bagaimana pengaruh perbandingan rasio data latih serta rasio data uji terhadap nilai akurasi yang dihasilkan. Tabel 4.26 merupakan perancangan perbandingan untuk rasio data latih serta rasio data uji. Pengujian rasio data latih dan data uji ada beberapa jenis antara lain :

1. 90% : 10% dimana nilainya dibulatkan menjadi 140 data latih dan 15 data uji.
2. 80% : 20% dimana nilainya dibulatkan menjadi 124 data latih dan 31 data uji.
3. 70% : 30% dimana nilainya dibulatkan menjadi 109 data latih dan 46 data uji.
4. 60% : 40% dimana nilainya dibulatkan menjadi 93 data latih dan 62 data uji.
5. 50% : 50% dimana nilainya dibulatkan menjadi 78 data latih dan 77 data uji.

Tabel 4.27 Perancangan Perbandingan Rasio Data Latih dan Data Uji

Percobaan ke-i	Pengujian ke 1	Pengujian ke 2	Pengujian ke 3	Pengujian ke 4	Pengujian ke 5
1					
2					
3					
4					
5					
Rata-rata					

4.5.2 Pengujian Pengaruh Hidden Neuron

Pengujian terhadap pengaruh jumlah *neuron hidden layer* dilakukan dengan mengubah setiap nilai dari *hidden neuron* dimana nilai *hidden neuron* sudah ditentukan terlebih dahulu. Nilai *hidden neuron* yang dipakai antara lain 2, 4, 6, 8, 10. Tabel 4.27 merupakan perancangan perbandingan nilai *hidden neuron.*

Tabel 4.28 Perancangan Perbandingan Jumlah *Hidden Neuron*

Percobaan ke-i	Pengujian ke 1	Pengujian ke 2	Pengujian ke 3	Pengujian ke 4	Pengujian ke 5
1					
2					
3					
4					
5					
Rata-rata					

4.5.3 Pengujian Perbandingan Fungsi Aktivasi

Pengujian dari fungsi aktivasi memiliki tujuan untuk melakukan perubahan pada nilai *input* menjadi suatu *output* tertentu. Pada penelitian yang dilakukan ini akan mempergunakan fungsi sigmoid biner, sigmoid bipolar, radial basis, linier serta sin. Tabel 4.28 merupakan perancangan perbandingan fungsi aktivasi.

Tabel 4.29 Perancangan Perbandingan Fungsi Aktivasi

Percobaan ke-i	Sigmoid biner	Sigmoid bipolar	Radial basis	Linier	sin
1					
2					
3					
4					
5					
Rata-rata					

BAB 5 IMPLEMENTASI

5.1 Batasan Implementasi

Dalam batasan implementasi yang akan digunakan untuk sistem klasifikasi gagal ginjal kronis menggunakan *Exteme Learning Machine* adalah seperti berikut ini:

1. Sistem yang dibangun menggunakan bahasa pemrograman *Java*.
2. Data yang akan digunakan langsung di inputkan manual ke dalam sistem.
3. Hasil keluaran yang dihasilkan merupakan hasil klasifikasi mengenai apakah penderita mengalami gagal ginjal kronis atau tidak mengalami gagal ginjal kronis.
4. Untuk data yang digunakan menggunakan 25 fitur termasuk kelas aslinya.
5. Dari ke 25 fitur tersebut fitur ke 1 sampai 24 namanya diubah menjadi x1 samapi x24 dan fitur ke 25 merupakan fitur kelas.

5.2 Implementasi Sistem

Untuk bagian implementasi sistem akan menjelaskan mengenai kode program yang akan digunakan dalam membuat perhitungan penyelesaian mengenai masalah klasifikasi gagal ginjal kronis menggunakan *Extreme Learning Machine*.

5.2.1 Implementasi Matriks Data Latih

Dalam pengimplementasian mengenai sistem, nilai dari data latih diambil langsung pada *file Microsoft Excel*. *File* dalam *excel* berisi *dataset* serta nilai matriks target. Dimana data set yang ada di *excel* nantinya akan dibagi menjadi data latih dan data uji, sedangkan matriks target akan digunakan untuk proses perhitungan dalam metode ELM. Sebelumnya nilai data set harus dinormalisasi supaya range antara setiap fitur memiliki kesamaan. Pada Kode Sumber 5.1 adalah inisialisasi mengenai normalisasi *dataset*.

```
1  public void normalisasidata() {
2  for (int i = 0; i < xawal.length; i++) {
3              for (int j = 0; j < xawal[0].length; j++) {
4                  normalisasi[i][j] = (xawal[i][j] - min[j]) /
5  (max[j] - min[j]);
6              }
7          }
}
```

Kode Sumber 5.1 Implementasi Normalisasi Data

Kode Sumber 5.1 menjelaskan mengenai normalisasi data:

Baris 1 - 7 : Proses pembacaan baris ke i dan kolom ke j, kemudian nilainya diambil dan dilakukan perhitungan normalisasi.

5.2.2 Implementasi Inisialisasi Nilai Bobot dan Bias

Dalam pengimplementasian inisialisasi dari nilai *weight* serta bias dilakukan secara *random.* Untuk ukuran ordo dari matriks *weight* akan dilakukan penyesuaian dengan nilai *neuron hidden layer* serta banyak fitur. Kemudian untuk ukuran dari ordo nilai matriks bias akan dilakukan penyesuaian dengan nilai jumlah data yang akan digunakan serta nilai *neuron hidden layer*. Kode Sumber 5.2 merupakan pengimplementasian nilai *weight* dan nilai bias dari sistem.

```
1  public void randomBobot() {
2          Random rand = new Random();
3          for (int i = 0; i < bobot.length; i++) {
4              for (int j = 0; j < bobot[0].length; j++) {
5                  bobot[i][j] = rand.nextInt(2001) - 1000;
6                  bobot[i][j] /= 1000;
7              }
8          }
9      }
10
11 public void randomBias() {
12         Random rand = new Random();
13         for (int i = 0; i < bias.length; i++) {
14             for (int j = 0; j < bias[0].length; j++) {
15                 bias[i][j] = rand.nextInt(101);
16                 bias[i][j] /= 1000;
17             }
18         }
19     }
```

Kode Sumber 5.2 Implementasi Inisialisasi Nilai Bobot dan Bias

Kode Sumber 5.2 menjelaskan mengenai nilai *weight* serta bias:

Baris 1 - 9 : Proses penginisialisasian *weight* secara *random* dengan nilai *range* -1 sampai 1.

Baris 11 – 19 : Proses penginisialisasian bias secara *random* dengan nilai *range* 0 sampai 1.

5.2.3 Implementasi Perhitungan Keluaran *Hidden Neuron*

Dalam implementasi perhitungan keluaran *hidden neuron* akan dilakukan sesuai dengan rumus dari metode ELM. Dimana untuk proses perhitungan akan mulai dengan menghitung perkalian matriks pada data latih dengan matriks bobot yang telah di *transpose.* Kemudian hasil dari perkalian tersebut akan dilakukan penjumlahan dengan nilai bias yang sudah tersedia. Kode Sumber 5.3 merupakan penginisialisasian perhitungan *output hidden neuron.*

```
1  public void bobotTranspose() {
2          for (int i = 0; i < bobotTranspose.length; i++) {
3              for (int j = 0; j < bobotTranspose[0].length;
4  j++) {
5                  bobotTranspose[i][j] = bobot[j][i];
6              }
7          }
8      }
9
```

```
10  public void hitungHinit() {
11  for (int i = 0; i < dataLatih.length; i++) {
12            for (int j = 0; j < jlhHiddenNeuron.length; j++) {
13                  for (int k = 0; k < normalisasi[0].length;
14  k++) {
15
16  Hinit[i][j]=Hinit[i][j]+(normalisasi[i][k]*bobotSementara[k
17  ][j]);
18                  }
19                  Hinit[i][j] = Hinit[i][j] +
20  biasSementara[i][j];
21              }
22          }
23  }
```

Kode Sumber 5.3 Implementasi Perhitungan Keluaran *Hidden Neuron*

Kode Sumber 5.3 menjelaskan mengenai perhitungan *output hidden neuron*:

Baris 1 - 7 : Proses perhitungan *transpose* nilai matriks *weight*.

Baris 9 – 23 : Proses perhitungan dari hasil perkalian matriks data latih dengan matriks *weight* yang telah di *transpose*. Kemudian hasilnya akan dilakukan penjumlah dengan nilai dari bias.

5.2.4 Implementasi Perhitungan Keluaran *Hidden Neuron* dengan Fungsi Aktivasi

Setelah didapatkan nilai keluran dari *hidden neuron,* langkah berikutnya yaitu menghitung nilai keluran dari *hidden neuron* dengan fungsi aktivasi. Dalam proses perhitungan nilai keluaran *hidden neuron* dengan fungsi aktivasi untuk proses *training* akan mengalami kesamaan dengan proses perhitungan nilai keluaran *hidden neuron* dengan fungsi aktivasi di proses *testing.* Kode Sumber 5.4 merupakan implementasi perhitungan *output hidden neuron* menggunakan fungsi aktivasi.

```
1   public void hitungSigmoidBiner() {
2   for (int i = 0; i < Hinit.length; i++) {
3             for (int j = 0; j < Hinit[0].length; j++) {
4                 if (jComboBox2.getSelectedIndex() == 0) {
5                       H[i][j] = 1 / (1 + (Math.pow(Math.E, (-1
6   * Hinit[i][j])));
7                       jTable6.setValueAt(i + 1, i, 0);
8                       jTable6.setValueAt((H[i][j]), i, j + 1);
9
10                      System.out.println("sigmoid biner");
11              } else if (jComboBox2.getSelectedIndex() == 1) {
12                      H[i][j] = (1 - (Math.pow(Math.E, (-1 *
13  Hinit[i][j]))) / (1 + (Math.pow(Math.E, (-1 *
14  Hinit[i][j]))));
15                      jTable6.setValueAt(i + 1, i, 0);
16                      jTable6.setValueAt((H[i][j]), i, j + 1);
17
18                      System.out.println("sigmoid bipolar");
19              } else if (jComboBox2.getSelectedIndex() == 2) {
20                      H[i][j] = (Math.E * (-1 *
21  (Math.pow(Hinit[i][j], 2))));
22                      jTable6.setValueAt(i + 1, i, 0);
```

```
23                      jTable6.setValueAt((H[i][j]), i, j + 1);
24
25                  System.out.println("sogmoid radial basis");
26              } else if (jComboBox2.getSelectedIndex() == 3) {
27                      H[i][j] = (Hinit[i][j]);
28                      jTable6.setValueAt(i + 1, i, 0);
29                      jTable6.setValueAt((H[i][j]), i, j + 1);
30
31                      System.out.println("sigmoid linier");
32              } else {
33                      H[i][j] = Math.sin(Hinit[i][j]);
34                      jTable6.setValueAt(i + 1, i, 0);
35                      jTable6.setValueAt((H[i][j]), i, j + 1);
36
37                      System.out.println("sigmoid sin");
38                  }
39              }
40          }
41      }
```

Kode Sumber 5.4 Implementasi Perhitungan Keluaran *Hidden Neuron* dengan Fungsi Aktivasi

Kode Sumber 5.4 menjelaskan mengenai perhitungan *output hidden neuron* menggunakan fungsi aktivasi:

Baris 4 - 10 : Proses untuk melakukan perhitungan fungsi aktivasi sigmoid biner.

Baris 11 - 18 : Proses untuk melakukan perhitungan fungsi aktivasi sigmoid bipolar.

Baris 19 - 25 : Proses untuk melakukan perhitungan fungsi aktivasi sigmoid radial basis.

Baris 26 - 31 : Proses untuk melakukan perhitungan fungsi aktivasi sigmoid linier.

Baris 32 - 38 : Proses untuk melakukan perhitungan fungsi aktivasi sigmoid sin.

5.2.5 Implementasi Perhitungan Moore-*Penrose*

Dalam perhitungan *Moore-Penrose* dilakukan berdasarkan rumus yang terdapat pada algoritma ELM. Perhitungan ini dilakukan hanya pada proses *training* sedangkan untuk proses *testing* perhitungan ini tidak perlu dilakukan. Kode Sumber 5.5 merupakan implementasi perhitungan *pseudo-inverse* dengan *moore-penrose*.

```
1   public void HTranspose() {
2   for (int i = 0; i < HTranspose.length; i++) {
3               for (int j = 0; j < HTranspose[0].length; j++) {
4                   HTranspose[i][j] = H[j][i];
5               }
6           }
7       }
8   public void hitungMoorePenrose() {
9           for (int i = 0; i < HtH.length; i++) {
10              for (int j = 0; j < HtH[0].length; j++) {
11                  for (int k = 0; k < H.length; k++) {
12                      HtH[i][j] += HTranspose[i][k] * H[k][j];
13                  }
```

```
14              }
15          }
16          for (int i = 0; i < HtH.length; i++) {
17              for (int j = 0; j < HtH[0].length; j++) {
18                  if (i == j) {
19                      invers[i][j] = 1;
20                  } else {
21                      invers[i][j] = 0;
22                  }
23              }
24          }
25          for (int i = 0; i < HtH.length; i++) {
26              double t = HtH[i][i];
27              for (int k = 0; k < HtH[0].length; k++) {
28                  invers[i][k] = invers[i][k] / t;
29                  HtH[i][k] = HtH[i][k] / t;
30              }
31
32              for (int j = 0; j < HtH[0].length; j++) {
33                  double c = HtH[j][i];
34                  for (int l = 0; l < HtH[0].length; l++) {
35                      if (i != j) {
36                invers[j][l] = invers[j][l] - c * invers[i][l];
37                HtH[j][l] = HtH[j][l] - c * HtH[i][l];
38                      }
39                  }
40              }
41          }
42          for (int i = 0; i < jlhHiddenNeuron.length; i++) {
43            for (int j = 0; j < dataLatih.length; j++) {
44              for (int k = 0; k < jlhHiddenNeuron.length; k++) {
45  moore_penrose[i][j]+=invers[i][k]*HTranspose[k][j];
46                  }
47              }
48          }
49  }
```

Kode Sumber 5.5 Implementasi Perhitungan *Pseudo-Inverse* dengan *Moore-Penrose*

Kode Sumber 5.5 menjelaskan mengenai perhitungan *Moore-Penrose*:

Baris 1 - 7 : Proses untuk melakukan *transpose* dari nilai hidden neuron menggunakan fungsi aktivasi.

Baris 9 - 16 : Proses untuk perhitungan antara perkalian dari matriks *hidden neuron* menggunakan fungsi aktivasi yang telah dilakukan *transpose* dengan matriks *hidden neuron* dengan fungsi aktivasi.

Baris 17 - 41 : Proses untuk perhitungan nilai *invers* hasil dari proses perkalian matriks *hidden neuron* menggunakan fungsi aktivasi yang telah dilakukan *transpose* dengan matriks keluaran *hidden neuron* dengan fungsi aktivasi.

Baris 42 – 49 : Proses untuk perhitungan *Moore-Penrose*.

5.2.6 Implementasi Perhitungan Bobot Keluaran

Dalam perhitungan bobot keluaran dilakukan berdasarkan rumus yang terdapat pada algoritma ELM. Perhitungan ini dilakukan hanya pada proses *training* sedangkan untuk proses *testing* perhitungan ini tidak perlu dilakukan. Kode Sumber 5.6 merupakan implementasi perhitungan bobot keluaran.

```
1  public void T_transpose() {
2          for (int i = 0; i < dataLatih.length; i++) {
3              for (int j = 0; j < 2; j++) {
4                  T_transpose[i][j] = Tkelass[j][i];
5              }
6          }
7      }
8
9  public void hitungBeta() {
10         for (int i = 0; i < jlhHiddenNeuron.length; i++) {
11             for (int j = 0; j < Tkelass.length; j++) {
12                 for (int k = 0; k < dataLatih.length; k++) {
13                     beta[i][j]      =      beta[i][j]      +
14 (moore_penrose[i][k] * T_transpose[k][j]);                 }
15             }
16         }
17     }
```

Kode Sumber 5.6 Implementasi Perhitungan Bobot Keluaran

Kode Sumber 5.6 menjelaskan mengenai perhitungan *output weight*:

Baris 1 - 7 : Proses untuk *transpose* matriks target.

Baris 9 - 17 : Proses untuk perhitungan bobot keluaran.

5.2.7 Implementasi Perhitungan Keluaran *Output Layer*

Dalam perhitungan keluran *output layer* dilakukan berdasarkan rumus yang terdapat pada algoritma ELM. Untuk proses *testing* nilai dari bobot keluaran yang dihasilkan dari proses *training* akan digunakan untuk melakukan perhitungan nilai keluaran *output layer*. Kode Sumber 5.7 merupakan implementasi perhitungan keluaran *output layer.*

```
1  public void HitungKeluaranOutputLayer() {
2  for (int i = 0; i < dataLatih.length; i++) {
3           for (int j = 0; j < Tkelass.length; j++) {
4             for (int k = 0; k < jlhHiddenNeuron.length; k++){
5  nilaiOutput[i][j]=nilaiOutput[i][j]+(H[i][k]*beta[k][j]);   }
6              }
7          }
8  }
```

Kode Sumber 5.7 Implementasi Perhitungan Keluaran *Output Layer*

Kode Sumber 5.7 menjelaskan mengenai perhitungan *output layer*:

Baris 1 - 8 : Proses untuk perhitungan matriks *output layer.*

5.2.8 Implementasi Matriks Data Uji

Dalam implementasi mengenai sistem, nilai data uji diambil langsung dari data set yang sebelumnya sudah terdapat dalam *excel*. Dimana dalam data set terdiri dari data uji dan nilai target. Sebelumnya nilai data set harus dinormalisasi supaya *range* antara setiap fitur memiliki kesamaan. Pada Kode Sumber 5.8 merupakan implementasi mengenai inisialisasi data uji.

```
1  public void dataUji() {
2  for (int i = 0; i < dataUji.length; i++) {
3             for (int j = 0; j < dataUji[0].length; j++) {
4                 dataUji[i][j]       =       normalisasi[i       +
5  (dataLatih.length)][j];           }
6          }
7    }
```

Kode Sumber 5.8 Implementasi Inisialisasi Data Uji

Penjelasan Kode Sumber 5.8 mengenai implementasi inisialisasi data uji sebagai berikut:

Baris 1 - 7 : Proses inisialisasi data uji.

5.2.9 Implementasi Inisialisasi Nilai Bias Data Uji

Dalam implementasi inisialisasi nilai bias dilakukan secara *random.* Untuk ukuran ordo dari matriks bias disesuaikan dengan jumlah data yang akan digunakan dan nilai *hidden neuron*. Kode Sumber 5.9 merupakan implementasi nilai bias dari sistem.

```
1  public void randomBiasDataUji() {
2          Random rand = new Random();
3          for (int i = 0; i < biasDataUji.length; i++) {
4              for (int j = 0; j < biasDataUji[0].length; j++) {
5                  biasDataUji[i][j] = rand.nextInt(101);
6                  biasDataUji[i][j] /= 1000;
7              }
8          }
```

Kode Sumber 5.9 Inisialisasi Nilai Bias Data Uji

Kode Sumber 5.9 menjelaskan mengenai nilai bias data uji:

Baris 1 - 8 : Proses untuk melakukan penginisialisasian bias data uji dengan cara *random* dengan nilai range 0 sampai 1.

5.2.10 Implementasi Perhitungan Keluaran *Hidden Neuron* Data Uji

Dalam implementasi perhitungan keluaran *hidden neuron* akan dilakukan sesuai dengan rumus dari metode ELM. Dimana untuk proses perhitungan akan mulai dengan menghitung perkalian matriks pada data uji dengan matriks bobot yang telah di *transpose.* Kemudian hasil dari perkalian tersebut akan dilakukan penjumlahan dengan nilai bias data uji yang sudah tersedia. Kode Sumber 5.10 merupakan implementasi *output hidden neuron.*

```
1  public void hitungHinitDataUji() {
2  for (int i = 0; i < dataUji.length; i++) {
3             for (int j = 0; j < jlhHiddenNeuron.length; j++) {
```

```
4                   for (int k = 0; k < dataUji[0].length; k++) {
5                       HinitDataUji[i][j] = HinitDataUji[i][j] +
6   (dataUji[i][k] * bobotTranspose[k][j]);                 }
7                   HinitDataUji[i][j]  =  HinitDataUji[i][j]  +
8   biasDataUji[i][j];            }
9           }
10  }
```

Kode Sumber 5.10 Implementasi Perhitungan Keluran *Hidden Neuron* Data Uji

Kode Sumber 5.10 mejelaskan mengenai keluaran *hidden neuron* data uji:

Baris 1 - 10 : Proses untuk perhitungan perkalian matriks data uji dengan matriks *weight* yang sudah dilakukan *transpose*. Kemudian hasil nilai dilakukan penjumlah dengan bias data uji.

5.2.11 Implementasi Perhitungan Keluaran *Hidden Neuron* dengan Fungsi Aktivasi

Setelah didapatkan nilai keluran dari *hidden neuron,* langkah berikutnya yaitu menghitung nilai keluran dari *hidden neuron* dengan fungsi aktivasi. Dalam proses perhitungan nilai keluaran *hidden neuron* dengan fungsi aktivasi untuk proses *testing* saat ini akan mengalami kesamaan dengan proses perhitungan nilai keluaran *hidden neuron* dengan fungsi aktivasi di proses *training*. Kode Sumber 5.11 merupakan implementasi *output hidden neuron* menggunakan fungsi aktivasi.

```
1   public void hitungSigmoidBinerDataUji() {
2     for (int i = 0; i < HDataUji.length; i++) {
3         for (int j = 0; j < HDataUji[0].length; j++) {
4             if (jComboBox2.getSelectedIndex() == 0) {
5                 HDataUji[i][j] = 1 / (1 + (Math.pow(Math.E, (-1
6   * HinitDataUji[i][j]))));
7                 jTable13.setValueAt(i + 1, i, 0);
8                 jTable13.setValueAt((HDataUji[i][j]), i, j + 1);
9
10                System.out.println("sigmoid biner data uji");
11            } else if (jComboBox2.getSelectedIndex() == 1) {
12                HDataUji[i][j] = (1 - (Math.pow(Math.E, (-1 *
13  HinitDataUji[i][j])))) / (1 + (Math.pow(Math.E, (-1 *
14  HinitDataUji[i][j]))));
15                jTable13.setValueAt(i + 1, i, 0);
16                jTable13.setValueAt((HDataUji[i][j]), i, j + 1);
17
18                System.out.println("sigmoid bipolar data uji");
19            } else if (jComboBox2.getSelectedIndex() == 2) {
20                HDataUji[i][j]    =    (Math.E    *    (-1    *
21  (Math.pow(HinitDataUji[i][j], 2))));
22                jTable13.setValueAt(i + 1, i, 0);
23                jTable13.setValueAt((HDataUji[i][j]), i, j + 1);
24
25             System.out.println("sogmoid radial basis data uji");
26            } else if (jComboBox2.getSelectedIndex() == 3) {
27                HDataUji[i][j] = (HinitDataUji[i][j]);
28                jTable13.setValueAt(i + 1, i, 0);
29                jTable13.setValueAt((HDataUji[i][j]), i, j + 1);
30
```

```
31                  System.out.println("sigmoid linier data uji");
32          } else {
33                  HDataUji[i][j] = Math.sin(HinitDataUji[i][j]);
34                  jTable13.setValueAt(i + 1, i, 0);
35                  jTable13.setValueAt((HDataUji[i][j]), i, j + 1);
36
37                  System.out.println("sigmoid sin data uji");
38          }
39        }
40     }
41   }
42
```

Kode Sumber 5.11 Implementasi Perhitungan Keluaran *Hidden Neuron* dengan Fungsi Aktivasi

Kode Sumber 5.11 menjelaskan mengenai *output hidden neuron* menggunakan fungsi aktivasi:

Baris 4 - 10 : Proses untuk melakukan perhitungan fungsi aktivasi sigmoid biner.

Baris 11 - 18 : Proses untuk melakukan perhitungan fungsi aktivasi sigmoid bipolar.

Baris 19 - 25 : Proses untuk melakukan perhitungan fungsi aktivasi sigmoid radial basis.

Baris 26 - 31 : Proses untuk melakukan perhitungan fungsi aktivasi sigmoid linier.

Baris 32 - 38 : Proses untuk melakukan perhitungan fungsi aktivasi sigmoid sin.

5.2.12 Implementasi Perhitungan Keluaran *Output Layer* Data Uji

Dalam perhitungan keluran *output layer* data uji dilakukan berdasarkan rumus yang terdapat pada algoritma ELM. Untuk proses *testing* nilai dari bobot keluaran yang dihasilkan dari proses *training* akan digunakan untuk melakukan perhitungan nilai keluaran *output layer* data uji. Kode Sumber 5.12 merupakan *output layer* data uji.

```
1  public void HitungKeluaranOutputLayerDataUji() {
2  for (int i = 0; i < dataUji.length; i++) {
3         for (int j = 0; j < Tkelass.length; j++) {
4             for (int k = 0; k < jlhHiddenNeuron.length; k++) {
5           nilaiOutputDataUji[i][j]=nilaiOutputDataUji[i][j]
6  + (HDataUji[i][k] * beta[k][j]);                }
7               }
8          }
9  }
```

Kode Sumber 5.12 Implementasi Perhitungan Keluaran *Output Layer* Data Uji

Kode Sumber 5.12 mejelaskasn mengenai *output layer* data uji:

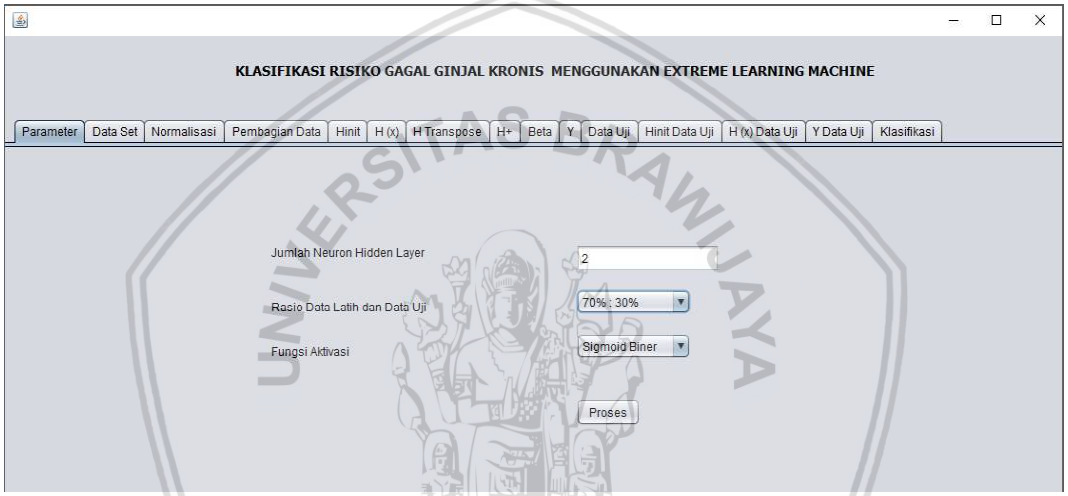
Baris 1 - 9 : Proses untuk perhitungan matriks *output layer* data uji.

5.3 Implementasi Antarmuka

Antarmuka dari sistem akan dipergunakan pengguna sistem dalam melakukan proses klasifikasi gagal ginjal kronis. Pada bagian antar muka sistem akan menyajikan semua halaman yang ada pada sistem.

5.3.1 Implementasi Antarmuka Halaman Parameter

Pada antarmuka parameter terdapat tiga parameter yaitu jumlah *neuron hidden layer,* rasio data latih dan data uji dan fungsi aktivasi. Ketiga parameter ini nantinya akan digunakan sebagai pengujian untuk metode *Extreme Learning Machine.* Dalam antarmuka parameter ada tombol *button* yang mempunyai fungsi melakukan proses selanjutnya untuk sistem. Gambar 5.1 merupakan implementasi antarmuka parameter.

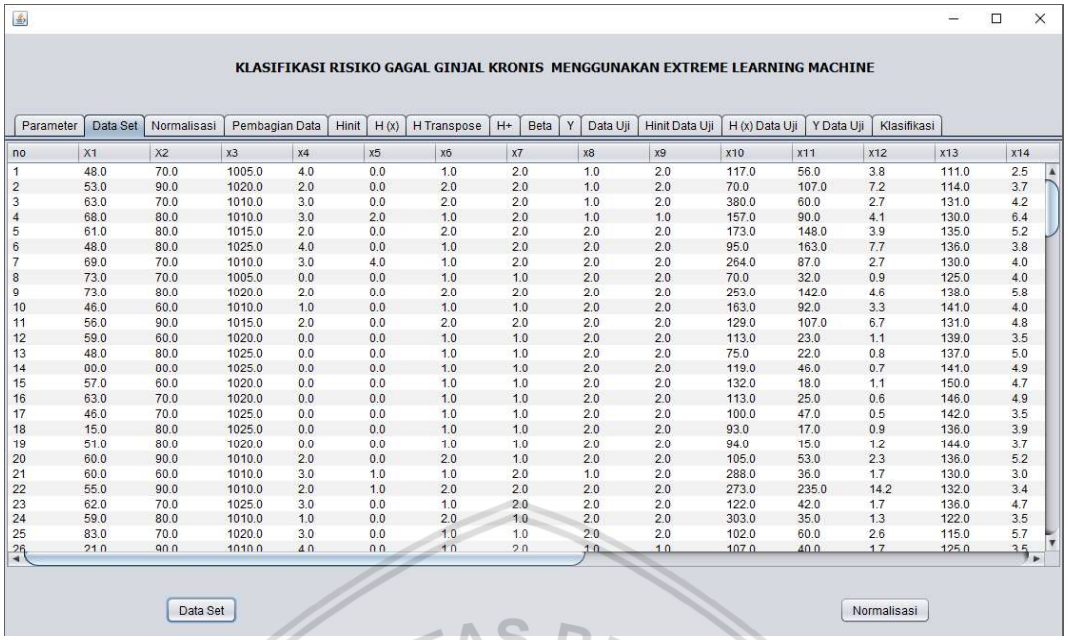


Gambar 5.1 Implementasi Antarmuka Halaman Parameter

Gambar 5.1 menerangkan mengenai antarmuka halaman pada parameter. Pada halaman ini nilai jumlah *neuron hidden layer,* rasio data latih dan data uji serta fungsi aktivasi akan ditentukan secara manual.

5.3.2 Implementasi Antarmuka Halaman *Dataset*

Dalam halaman antarmuka *dataset* terdapat *dataset* yang nantinya akan digunakan untuk proses klasifikasi gagal ginjal kronis menggunakan *Extreme Learning Machine.* Untuk data set sendiri terdiri dari 155 data yang memiliki 25 fitur termasuk kelas. Dalam antarmuka *dataset* terdapat tombol *button* yang berfungsi untuk melakukan proses selanjutnya untuk sistem dan tombol *dataset* yang berfungsi untuk melakukan proses *input* data yang ada di *excel*. Gambar 5.2 merupakan antarmuka *dataset*.



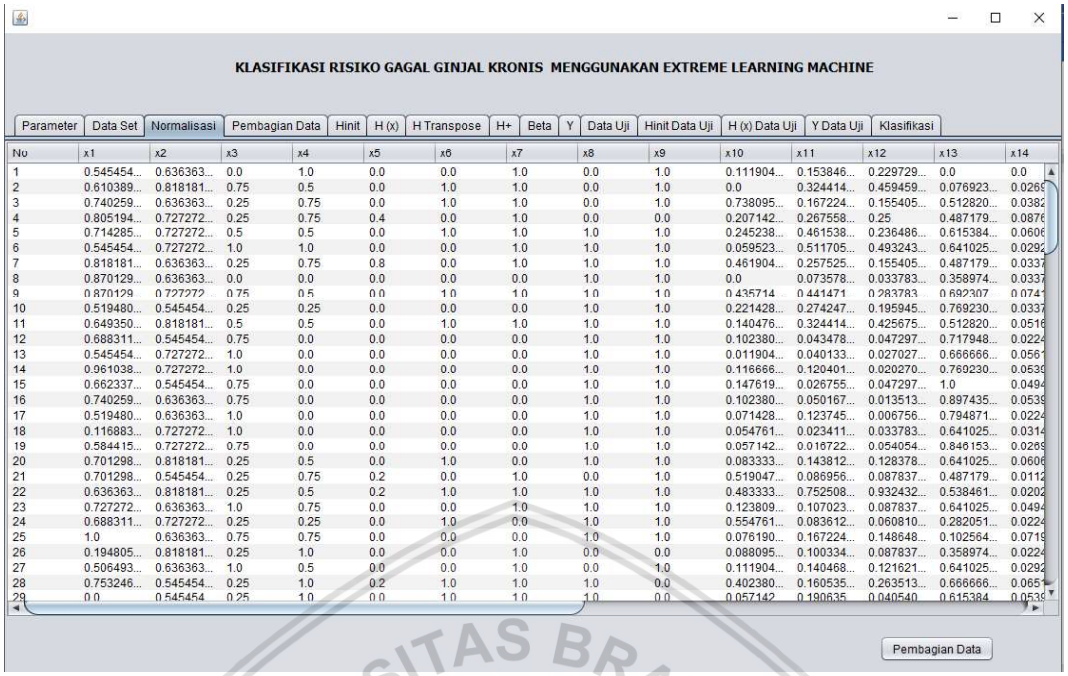
no	X1	X2	X3	x4	x5	x6	x7	x8	x9	x10	x11	x12	x13	x14
1	48.0	70.0	1005.0	4.0	0.0	1.0	2.0	1.0	2.0	117.0	56.0	3.8	111.0	2.5
2	53.0	90.0	1020.0	2.0	0.0	2.0	2.0	1.0	2.0	70.0	107.0	7.2	114.0	3.7
3	63.0	70.0	1010.0	3.0	0.0	2.0	2.0	1.0	2.0	380.0	60.0	2.7	131.0	4.2
4	68.0	80.0	1010.0	3.0	2.0	1.0	2.0	1.0	1.0	157.0	90.0	4.1	130.0	6.4
5	61.0	80.0	1015.0	2.0	0.0	2.0	2.0	2.0	2.0	173.0	148.0	3.9	135.0	5.2
6	48.0	80.0	1025.0	4.0	0.0	1.0	2.0	2.0	2.0	95.0	163.0	7.7	136.0	3.8
7	69.0	70.0	1010.0	3.0	4.0	1.0	2.0	2.0	2.0	264.0	87.0	2.7	130.0	4.0
8	73.0	70.0	1005.0	0.0	0.0	1.0	1.0	2.0	2.0	70.0	32.0	0.9	125.0	4.0
9	73.0	80.0	1020.0	2.0	0.0	2.0	2.0	2.0	2.0	253.0	142.0	4.6	138.0	5.8
10	46.0	60.0	1010.0	1.0	0.0	1.0	1.0	2.0	2.0	163.0	92.0	3.3	141.0	4.0
11	56.0	90.0	1015.0	2.0	0.0	2.0	2.0	2.0	2.0	129.0	107.0	6.7	131.0	4.8
12	59.0	60.0	1020.0	0.0	0.0	1.0	1.0	2.0	2.0	113.0	23.0	1.1	139.0	3.5
13	48.0	80.0	1025.0	0.0	0.0	1.0	1.0	2.0	2.0	75.0	22.0	0.8	137.0	5.0
14	00.0	00.0	1025.0	0.0	0.0	1.0	1.0	2.0	2.0	119.0	46.0	0.7	141.0	4.9
15	57.0	60.0	1020.0	0.0	0.0	1.0	1.0	2.0	2.0	132.0	18.0	1.1	150.0	4.7
16	63.0	70.0	1020.0	0.0	0.0	1.0	1.0	2.0	2.0	113.0	25.0	0.6	146.0	4.9
17	46.0	70.0	1025.0	0.0	0.0	1.0	1.0	2.0	2.0	100.0	47.0	0.5	142.0	3.5
18	15.0	80.0	1025.0	0.0	0.0	1.0	1.0	2.0	2.0	93.0	17.0	0.9	136.0	3.9
19	51.0	80.0	1020.0	0.0	0.0	1.0	1.0	2.0	2.0	94.0	15.0	1.2	144.0	3.7
20	60.0	90.0	1010.0	2.0	0.0	2.0	1.0	2.0	2.0	105.0	53.0	2.3	136.0	5.2
21	60.0	60.0	1010.0	3.0	1.0	1.0	2.0	1.0	2.0	288.0	36.0	1.7	130.0	3.0
22	55.0	90.0	1010.0	2.0	1.0	2.0	2.0	2.0	2.0	273.0	235.0	14.2	132.0	3.4
23	62.0	70.0	1025.0	3.0	0.0	1.0	2.0	2.0	2.0	122.0	42.0	1.7	136.0	4.7
24	59.0	80.0	1010.0	1.0	0.0	2.0	1.0	2.0	2.0	303.0	35.0	1.3	122.0	3.5
25	83.0	70.0	1020.0	3.0	0.0	1.0	1.0	2.0	2.0	102.0	60.0	2.6	115.0	5.7
26	21.0	90.0	1010.0	4.0	0.0	1.0	2.0	1.0	1.0	107.0	40.0	1.7	125.0	3.5

Gambar 5.2 Implementasi Antarmuka Halaman *Dataset*

Gambar 5.2 menjelaskan mengenai antarmuka halaman awal *dataset* pada sistem. *Dataset* tersebut terdiri dari 155 data yang nantinya akan dilakukan pembagian yaitu sebagain digunakan data latih dan sebagain digunakan data uji.

5.3.3 Implementasi Antarmuka Halaman *Normalisasi*

Dalam halaman antarmuka halaman *normalisasi* nantinya data dari data set pada halaman sebelumnya akan dilakukan proses *normalisasi* data. *Normalisasi* ini dilakukan supaya *range* antara nilai dari setiap fitur tidak terlalu jauh yaitu mempunyai range antara 0 sampai 1. Dalam proses *normalisasi* pada sistem menggunakan metode *normalisasi MaxMin.* Dalam antarmuka normalisasi terdapat tombol *button* yang berfungsi untuk melakukan proses selanjutnya untuk sistem. Gambar 5.3 merupakan antarmuka *normalisasi*.

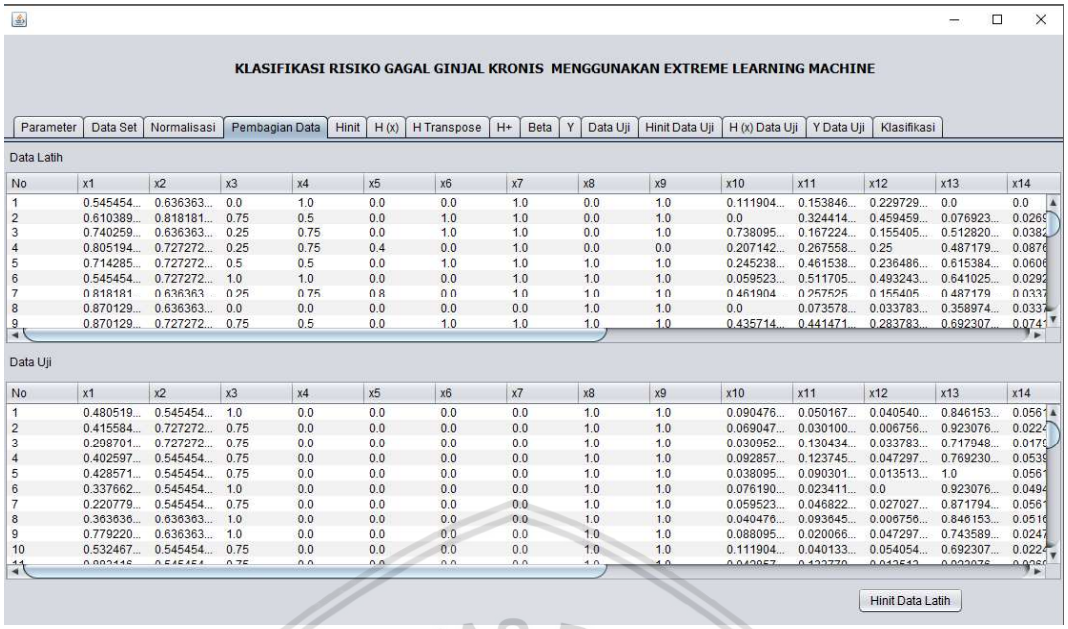


Gambar 5.3 Implementasi Antarmuka Halaman *Normalisasi*

Gambar 5.3 menjelaskan mengenai antarmuka *normalisasi.* Data yang digunakan dalam melakukan proses *normalisasi* yaitu menggunakan data dari data set pada halaman sebelumnya.

5.3.4 Implementasi Antarmuka Halaman Pembagian Data

Dalam halaman antarmuka pembagian data nantinya data dari halaman *normalisasi* akan dibagi menjadi 2 bagian. Bagian tersebut antara lain yaitu data yang nantinya digunakan untuk data latih dan data yang nantinya digunakan untuk data uji. Dalam antarmuka halaman pembagain data terdapat tombol *button* yang berfungsi untuk melakukan proses selanjutnya untuk sistem. Gambar 5.4 merupakan antarmuka pembagain data.

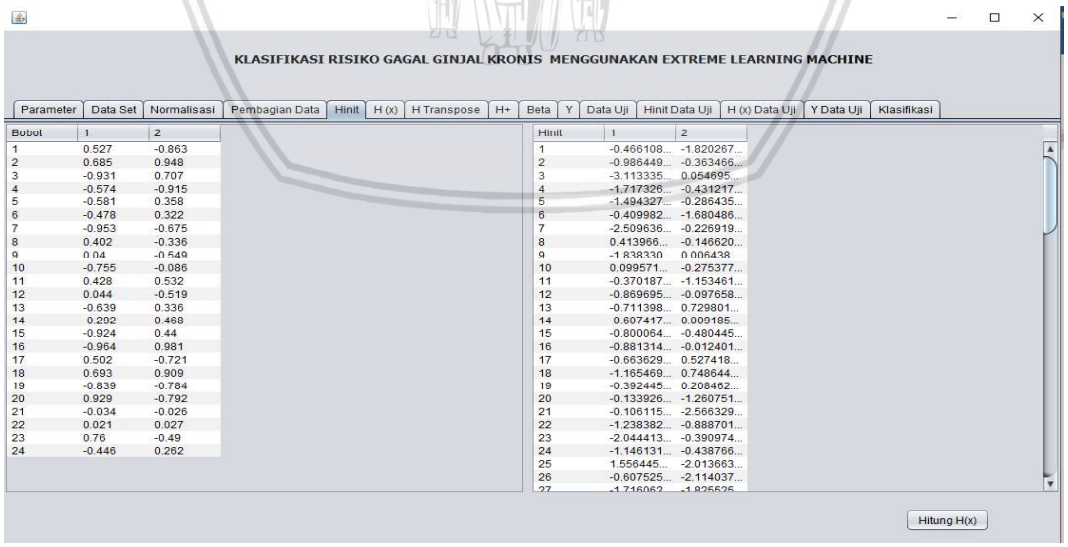


Gambar 5.4 Implementasi Antarmuka Halaman Pembagian Data

Gambar 5.4 menjelaskan mengenai antarmuka pembagian data. Data yang digunakan dalam melakukan proses pembagian data yaitu menggunakan data dari data yang sudah dinormalisasi pada halaman sebelumnya.

5.3.5 Implementasi Antarmuka Halaman Hinit

Dalam halaman antarmuka Hinit nantinya data dari halaman pembagian data yang bagian data latih akan diproses untuk mencari nilai Hinit. Dalam antarmuka halaman Hinit terdapat tombol *button* yang berfungsi untuk melakukan proses selanjutnya untuk sistem. Gambar 5.5 merupakan antarmuka *hidden neuron*.

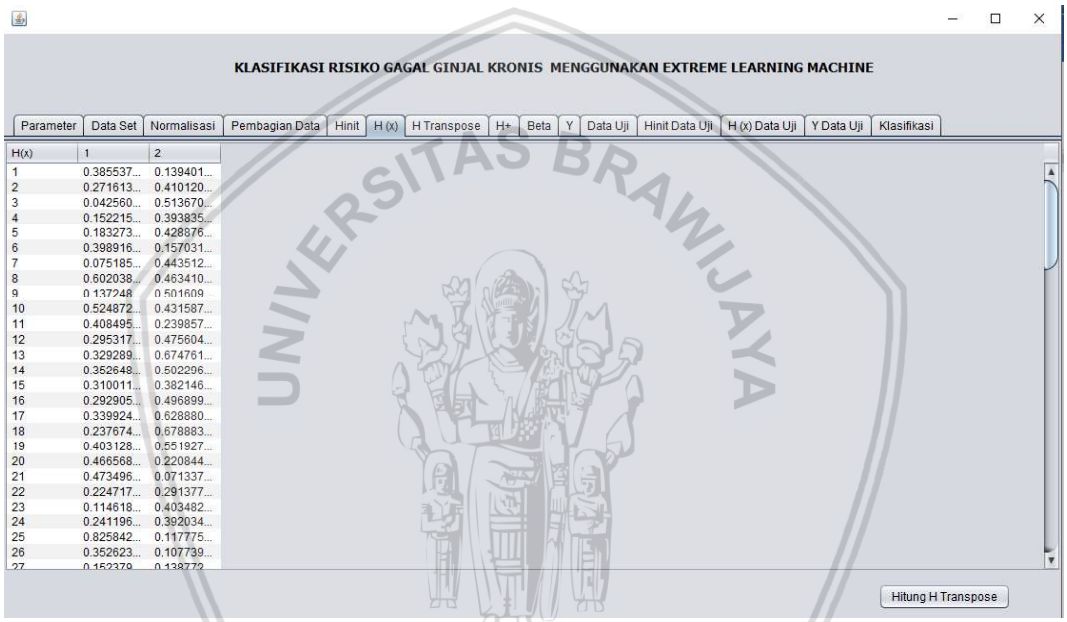


Gambar 5.5 Implementasi Antarmuka Halaman Hinit

Gambar 5.5 menjelaskan mengenai antarmuka Hinit. Data yang dipergunakan dalam melakukan perhitungan Hinit yaitu data dari data latih yang terdapat pada halaman pembagian data.

5.3.6 Implementasi Antarmuka Halaman Hinit dengan Fungsi Aktivasi Sigmoid Biner

Dalam halaman antarmuka Hinit dengan fungsi aktivasi sigmoid biner nantinya data dari halaman Hinit akan diproses untuk mencari nilai dari Hinit dengan fungsi aktivasi sigmoid biner. Dalam antarmuka halaman Hinit dengan fungsi aktivasi sigmoid biner terdapat tombol *button* yang berfungsi untuk melakukan proses selanjutnya untuk sistem. Gambar 5.6 merupakan implementasi antarmuka halaman Hinit dengan fungsi aktivasi sigmoid biner.

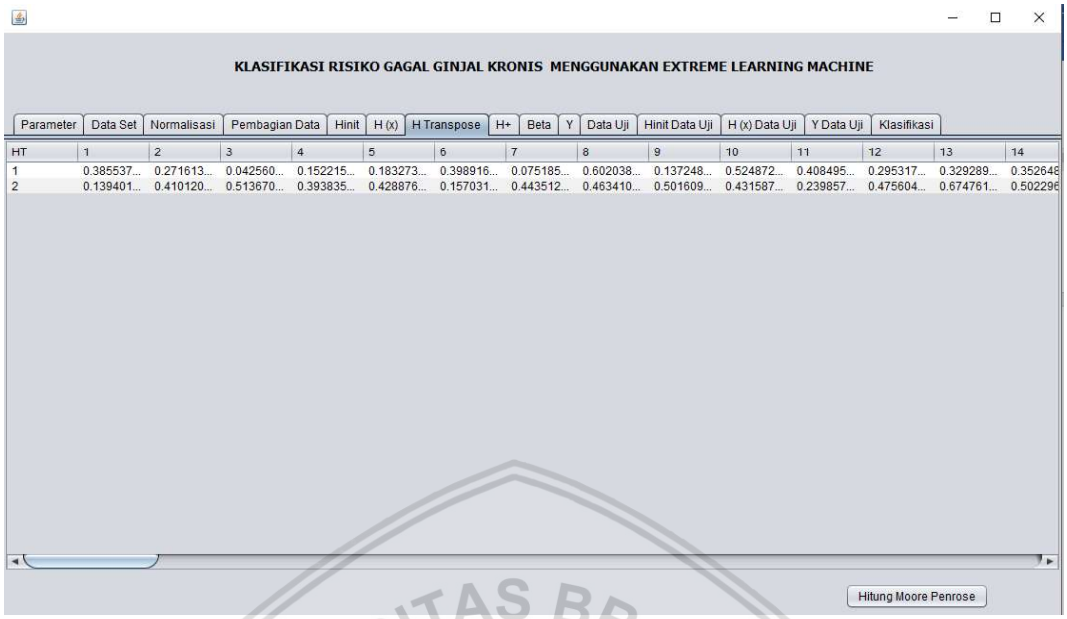


Gambar 5.6 Implementasi Antarmuka Halaman Hinit dengan Fungsi Aktivasi Sigmoid Biner

Gambar 5.6 menjelaskan mengenai implementasi antarmuka halaman Hinit dengan fungsi aktivasi sigmoid biner. Data yang digunakan dalam melakukan proses perhitungan Hinit dengan fungsi aktivasi sigmoid biner yaitu menggunakan data dari data Hinit yang terdapat pada halaman Hinit.

5.3.7 Implementasi Antarmuka Halaman Hinit dengan Fungsi Aktivasi Sigmoid Biner yang di *Transpose*

Dalam halaman antarmuka *HTranspose* nantinya data dari halaman Hinit dengan fungsi aktivasi sigmoid biner akan diproses untuk mencari nilai dari *HTranspose.* Dalam antarmuka halaman *HTransposes* terdapat tombol *button* yang berfungsi untuk melakukan proses selanjutnya untuk sistem. Gambar 5.7 merupakan antarmuka *HTranspose*.

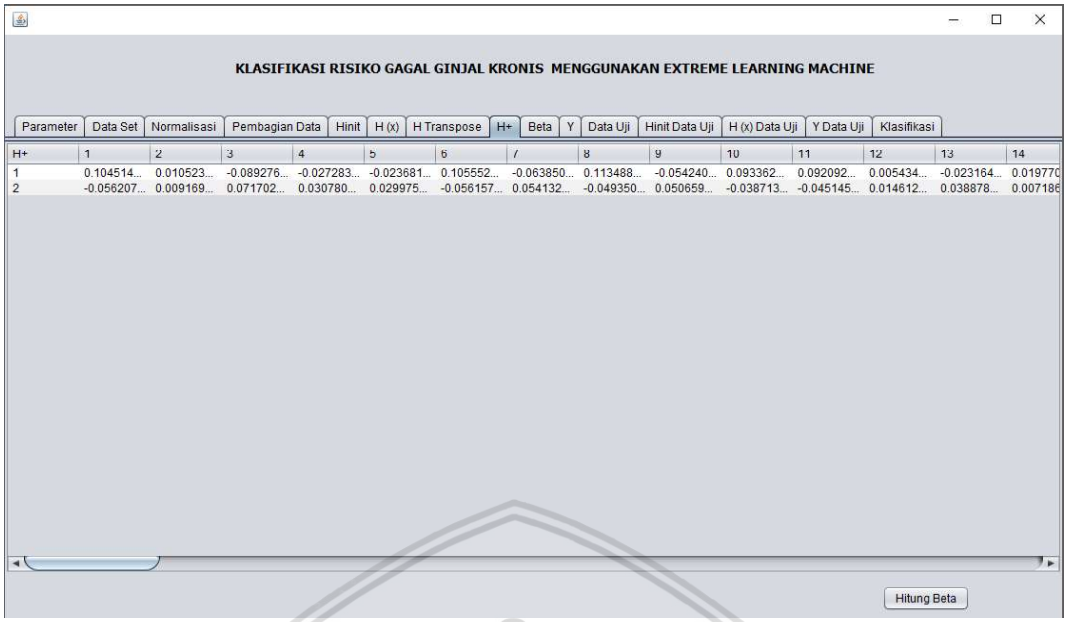


Gambar 5.7 Implementasi Antarmuka Halaman *Htranspose*

Gambar 5.7 menjelaskan mengenai antarmuka *Htranspose.* Data yang dipergunakan dalam melakukan perhitungan *HTranspose* yaitu dari dari data Hinit dengan fungsi aktivasi sigmoid biner yang terdapat pada halaman sebelumnya.

5.3.8 Implementasi Antarmuka Halaman *Pseudo-Inverse* dengan *Moore-Penrose*

Dalam halaman antarmuka *Pseudo-Inverse* dengan *Moore-Penrose* nantinya data dari halaman *HTranspose* akan diproses untuk mencari nilai dari *Pseudo-Inverse* dengan *Moore-Penrose*. Dalam antarmuka halaman *Pseudo-Inverse* dengan *Moore-Penrose* terdapat tombol *button* yang berfungsi untuk melakukan proses selanjutnya untuk sistem. Gambar 5.8 merupakan implementasi antarmuka halaman *Pseudo-Inverse* dengan *Moore-Penrose*.

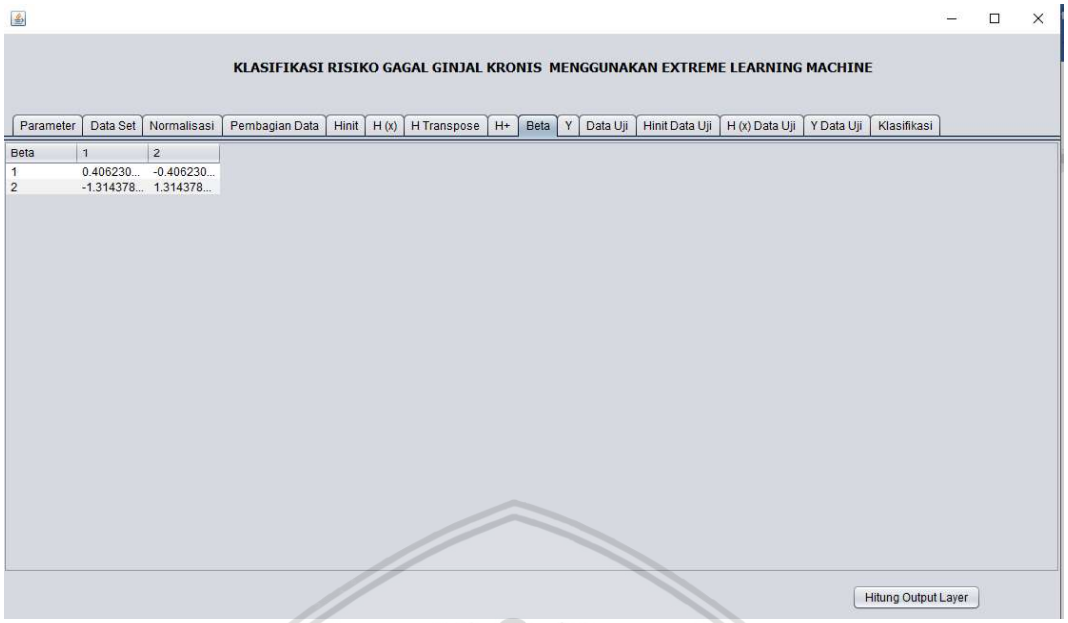


Gambar 5.8 Implementasi Antarmuka Halaman *Pseudo-Inverse* dengan *Moore-Penrose*

Gambar 5.8 menjelaskan mengenai implementasi antarmuka halaman *Pseudo-Inverse* dengan *Moore-Penrose*. Data yang digunakan dalam melakukan proses perhitungan *Pseudo-Inverse* dengan *Moore-Penrose* yaitu menggunakan data dari data *HTranspose* yang terdapat pada halaman sebelumnya.

5.3.9 Implementasi Antarmuka Halaman Bobot Keluaran

Dalam halaman antarmuka bobot keluaran nantinya data dari halaman *Pseudo-Inverse* dengan *Moore-Penrose* akan diproses untuk mencari nilai dari bobot keluaran. Dalam antarmuka halaman bobot keluaran terdapat tombol *button* yang berfungsi untuk melakukan proses selanjutnya untuk sistem. Gambar 5.9 merupakan antarmuka bobot keluaran.

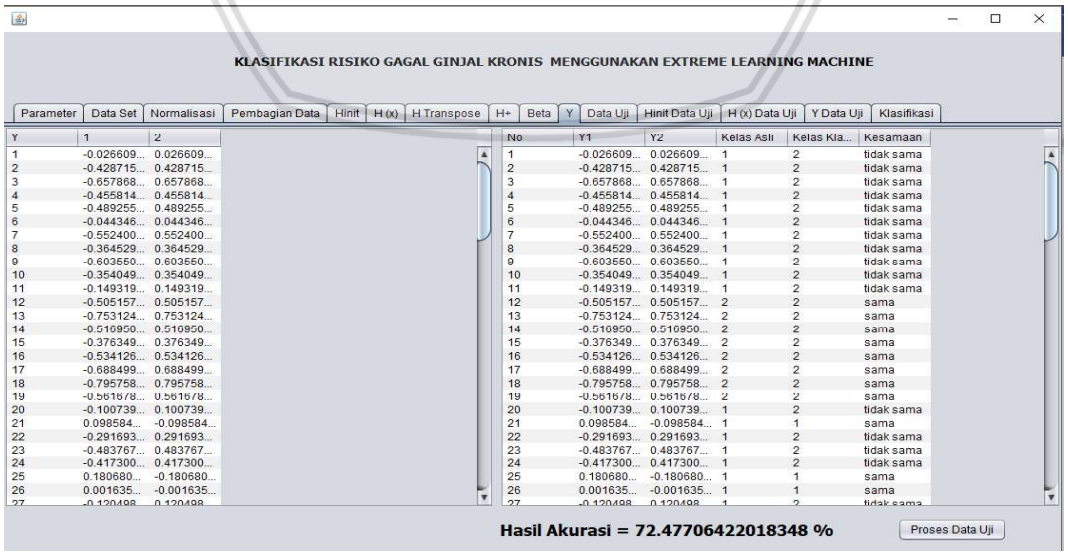


Gambar 5.9 Implementasi Antarmuka Halaman Bobot Keluaran

Gambar 5.9 menjelaskan mengenai antarmuka bobot keluaran. Data yang digunakan dalam melakukan proses perhitungan bobot keluaran yaitu menggunakan data dari data *Pseudo-Inverse* dengan *Moore-Penrose* yang terdapat pada halaman sebelumnya.

5.3.10 Implementasi Antarmuka Halaman *Output Layer*

Dalam halaman antarmuka *output layer* nantinya data dari halaman bobot keluaran akan diproses untuk mencari nilai dari *output layer.* Dalam antarmuka halaman *output layer* terdapat tombol *button* yang berfungsi untuk melakukan proses selanjutnya untuk sistem. Gambar 5.10 merupakan antarmuka *output layer*.

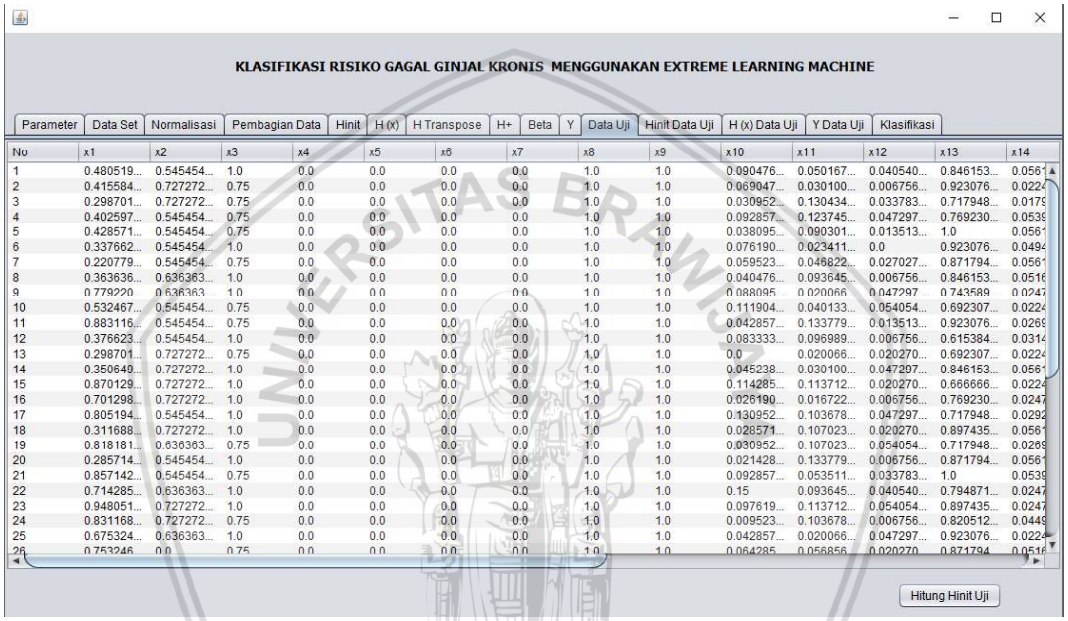


Gambar 5.10 Implementasi Antarmuka Halaman *Output Layer*

Gambar 5.10 menjelaskan mengenai antarmuka *output layer.* Data yang digunakan dalam melakukan proses perhitungan *output layer* yaitu menggunakan data dari data bobot keluaran yang terdapat pada halaman sebelumnya.

5.3.11 Implementasi Antarmuka Halaman Data Uji

Dalam halaman antarmuka data uji mengambil data dari halaman pembagian data yang bagain data uji. Nantinya data uji ini akan dilakukan pemrosesan untuk mendapatkan hasil dari klasifikasi gagal ginjal kronis menggunakan *Extreme Learning Machine.* Dalam antarmuka halaman data uji terdapat tombol *button* yang berfungsi untuk melakukan proses selanjutnya untuk sistem. Gambar 5.11 merupakan antarmuka data uji.

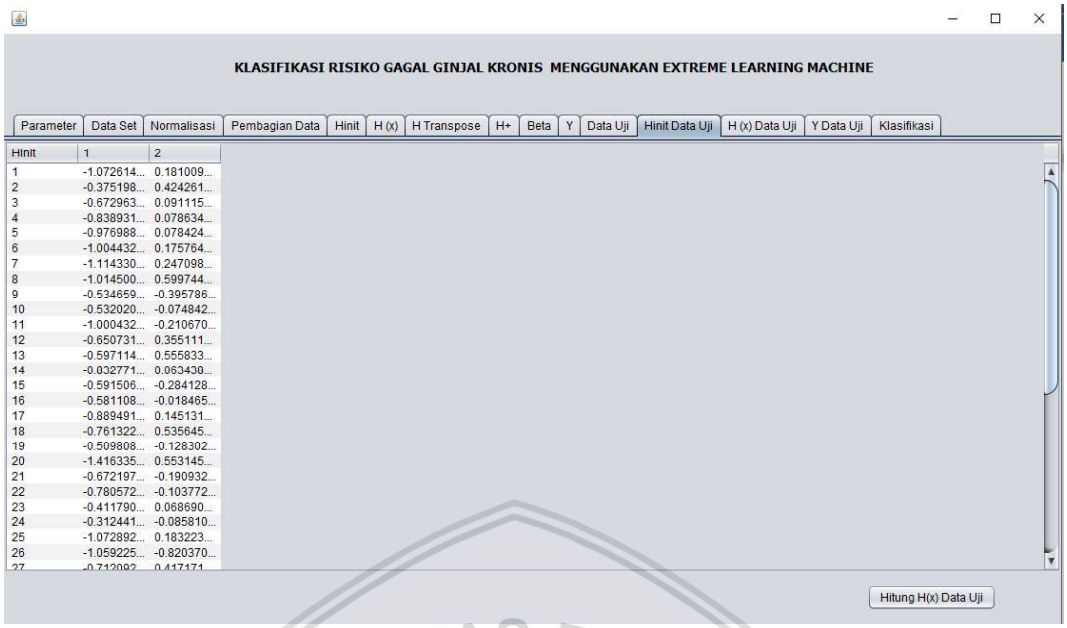


Gambar 5.11 Implementasi Antarmuka Halaman Data Uji

Gambar 5.11 menjelaskan mengenai antarmuka data uji. Nilai data dari halaman data uji ini diambil dari nilai data pada halaman pembagian data yang bagian data uji.

5.3.12 Implementasi Antarmuka Halaman *Hidden Neuron* Data Uji

Dalam halaman antarmuka *hidden neuron* data uji nantinya data dari halaman data uji akan diproses untuk mencari nilai dari *hidden neuron* data uji. Dalam antarmuka halaman *hidden neuron* data uji terdapat tombol *button* yang berfungsi untuk melakukan proses selanjutnya untuk sistem. Gambar 5.12 merupakan implementasi antarmuka halaman *hidden neuron* data uji.

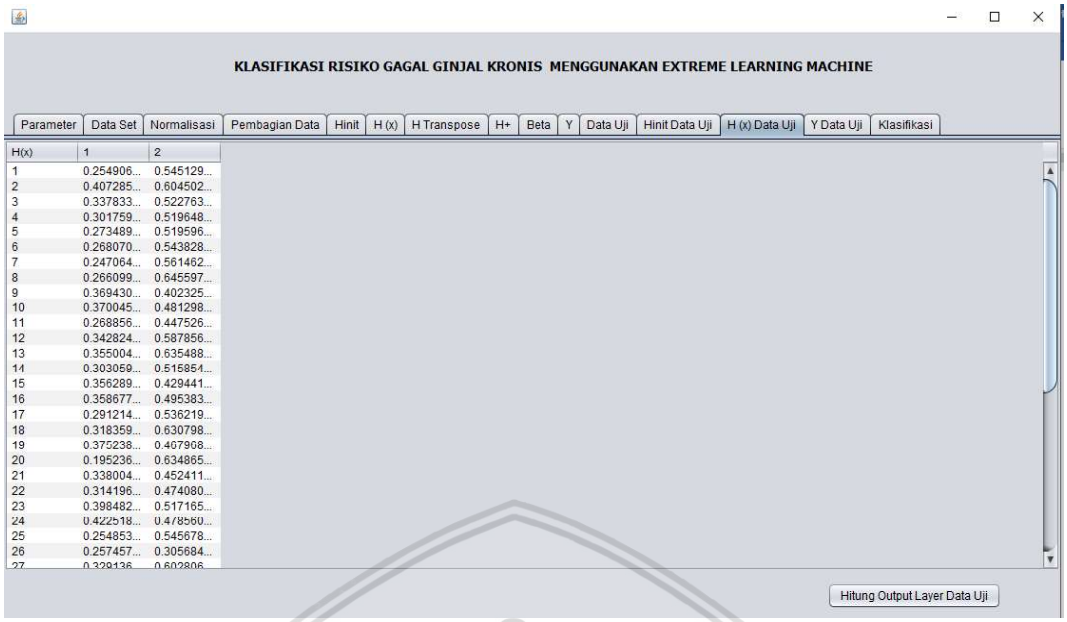


Gambar 5.12 Implementasi Antarmuka Halaman *Hidden Neuron* Data Uji

Gambar 5.12 menjelaskan mengenai implementasi antarmuka halaman *hidden neuron* data uji. Data yang digunakan dalam melakukan proses perhitungan *hidden neuron* data uji yaitu menggunakan data dari data uji yang terdapat pada halaman data uji.

5.3.13 Implementasi Antarmuka Halaman *Hidden Neuron* dengan Fungsi Aktivasi Sigmoid Biner Data Uji

Dalam halaman antarmuka *hidden neuron* dengan fungsi aktivasi sigmoid biner data uji nantinya data dari halaman *hidden neuron* data uji akan diproses untuk mencari nilai dari *hidden neuron* dengan fungsi aktivasi sigmoid biner data uji. Dalam antarmuka halaman *hidden neuron* dengan fungsi aktivasi sigmoid biner data uji terdapat tombol *button* yang berfungsi untuk melakukan proses selanjutnya untuk sistem. Gambar 5.13 merupakan implementasi antarmuka halaman *hidden neuron* dengan fungsi aktivasi sigmoid biner data uji.

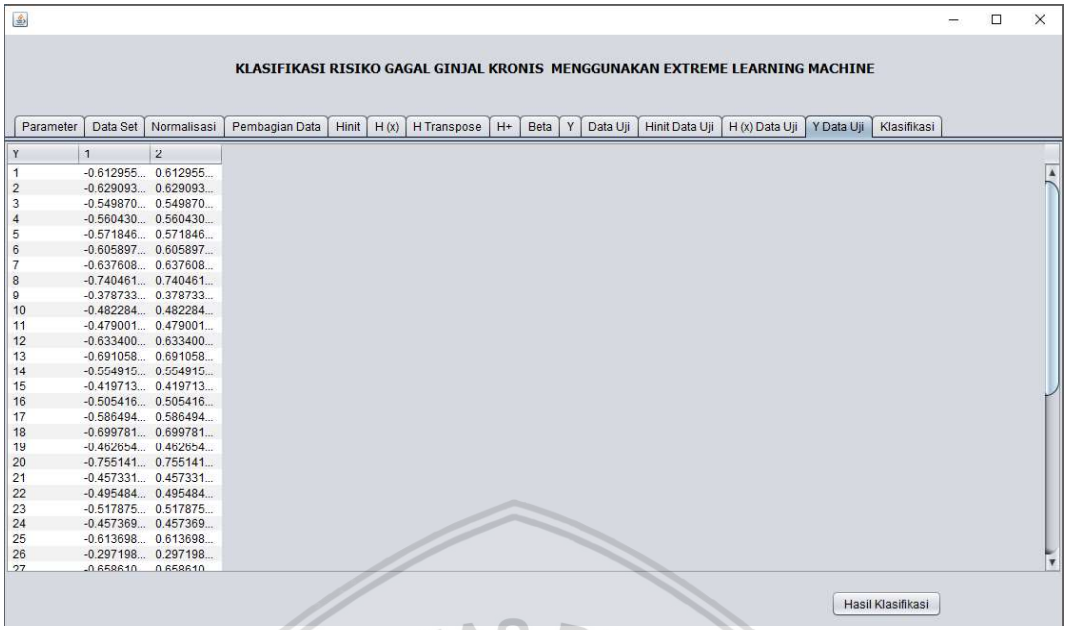


Gambar 5.13 Implementasi Antarmuka Halaman *Hidden Neuron* dengan Fungsi Aktivasi Sigmoid Biner Data Uji

Gambar 5.13 menjelaskan mengenai implementasi antarmuka halaman *hidden neuron* dengan fungsi aktivasi sigmoid biner data uji. Data yang digunakan dalam melakukan proses perhitungan *hidden neuron* dengan fungsi aktivasi sigmoid biner data uji yaitu menggunakan data dari data *hidden neuron* yang terdapat pada halaman *hidden neuron* data uji.

5.3.14 Implementasi Antarmuka Halaman *Output Layer* Data Uji

Dalam halaman antarmuka *output layer* data uji nantinya akan dilakukan perhitungan antara *hidden neuron* menggunakan fungsi aktivasi dikalikan dengan nilai bobot *output* dari hasil pada proses *training*. Dalam antarmuka halaman *output layer* terdapat tombol *button* yang berfungsi untuk melakukan proses selanjutnya untuk sistem. Gambar 5.14 merupakan implementasi antarmuka halaman *output layer* data uji.

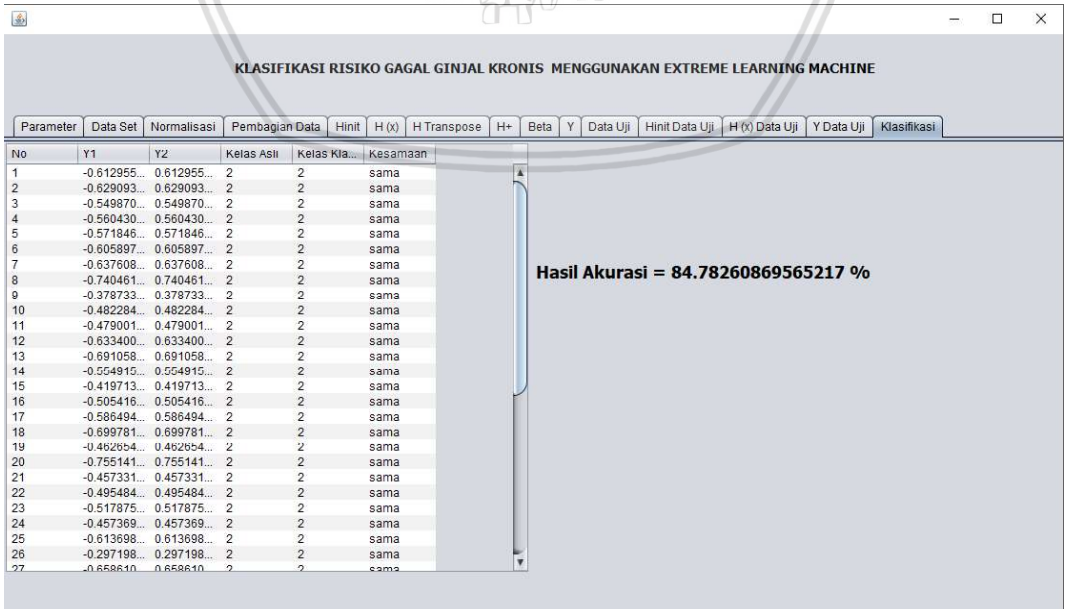


Gambar 5.14 Implementasi Antarmuka Halaman *Output Layer*

Gambar 5.14 menjelaskan mengenai antarmuka *output layer* data uji. Data yang digunakan dalam melakukan proses perhitungan *output layer* data uji yaitu menggunakan data dari data dari hasil proses nilai *hidden neuron* dengan fungsi aktivasi sigmoid biner data uji dikalikan dengan hasil proses nilai bobot *output* pada proses *training*.

5.3.15 Implementasi Antarmuka Halaman Hasil Klasifikasi

Halaman hasil klasifikasi adalah halaman yang mempunyai fungsi untuk memperlihatkan nilai hasil dari proses klasifikasi. Gambar 5.15 merupakan hasil dari proses klasifikasi gagal ginjal kronis menggunakan *Extreme Learning Machine*.



Gambar 5.15 Implementasi Antarmuka Halaman Hasil Klasifikasi

Gambar 5.15 menerangkan mengenai hasil dari proses klasifikasi. Hasil dari klasifikasi menggunakan metode *Extreme Learning Machine* ini akan dicocokkan dengan nilai kelas asli. Apabila nilai kelas asli sama dengan nilai kelas hasil klasifikasi maka pada kolom keterangan akan muncul nilai sama dan sebaliknya apabila nilai kelas asli tidak sama dengan nilai kelas hasil klasifikasi maka pada kolom keterangan akan muncul nilai tidak sama.

BAB 6 PENGUJIAN DAN ANALISIS

Pada bagian pengujian dan analisis membahas mengenai proses pengujian sistem yang telah dibangun dengan algoritma *Extreme Learning Machine.* Untuk proses pengujian sendiri akan dilakukan sesuai dengan rancangan yang sudah dibuat pada tahap sebelumnya. Proses untuk pengujian itu sendiri seperti bebarapa hal yaitu pengujian untuk perbandingan dari rasio data latih serta rasio data uji, pengujian untuk pengaruh *neuron hidden layer,* pengujian untuk perbandingan fungsi aktivasi.

6.1 Pengujian dan Analisis Perbandingan Rasio Data Latih dan Data Uji

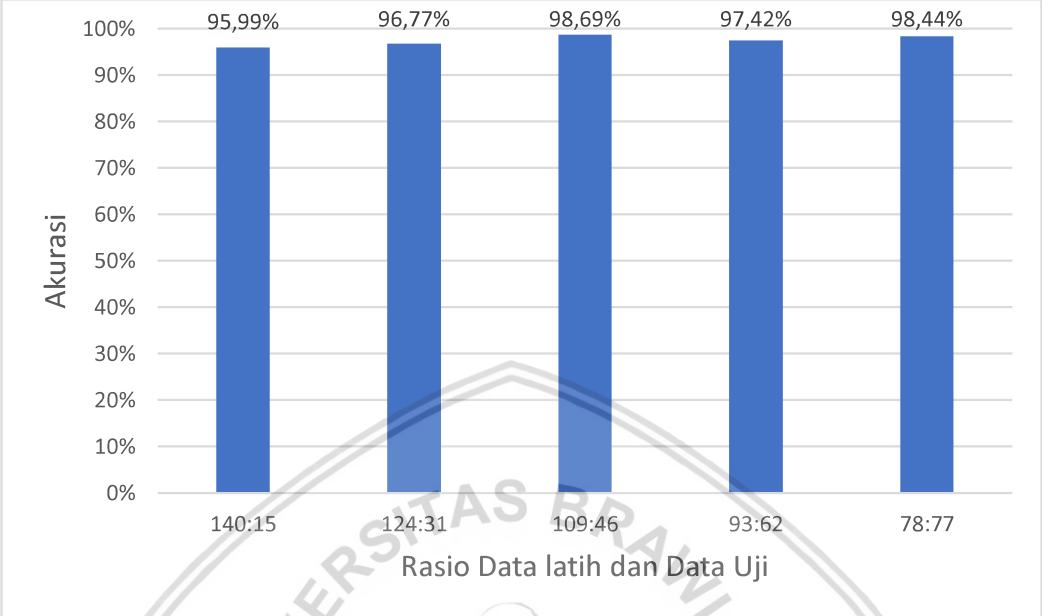
Pengujian untuk perbandingan rasio antara data latih serta data uji dilakukan bertujuan mengetahui seberapa besar pengaruh rasio antara data latih serta data uji terhadap nilai dari akurasi. Dalam pengujian ini nilai data latih maupun nilai dari data uji sudah ditentukan terlebih dahulu. Nilai data latih dan nilai data uji yang akan digunakan untuk pengujian dan analisis yaitu sebesar 90% : 10% dimana nilainya dibulatkan menjadi 140 data latih dan 15 data uji, 80% : 20% dimana nilainya dibulatkan menjadi 124 data latih dan 31 data uji, 70% : 30% dimana nilainya dibulatkan menjadi 109 data latih dan 46 data uji, 60% : 40% dimana nilainya dibulatkan menjadi 93 data latih dan 62 data uji, dan 50% : 50% dimana nilainya dibulatkan menjadi 78 data latih dan 77 data uji. Parameter pada pengujian ini nantinya akan menggunakan nilai neuron hidden layer berjumlah 10, nilai masukan sebanyak 24 buah, dan fungsi aktivasi yang digunakan adalah sigmoid biner. Pengujian akan dilakukan sebanyak 5 kali untuk setiap rasio.

Dalam melakukan proses perhitungan dengan menggunakan algoritme *Extreme Learning Machine* akan menghasilkan nilai yang berbeda dari setiap percobaan. Semua ini terjadi karena nilai dari bobot dan nilai bias yang diinisialisasikan secara *random*. Sehingga untuk mendapatkan nilai rata-rata keseluruhan harus melakukan 5 kali percobaan setiap rasionya. Tabel 6.1 merupakan pengujian perbandingan data latih serta data uji.

Tabel 6.1 Hasil Pengujian Perbandingan Data Latih dan Data Uji

Percobaan ke-i	90%:10%	80%:20%	70%:30%	60%:40%	50%:50%
	140:15	124:31	109:46	93:62	78:77
1	93,33%	96,77%	95,65%	98,39%	97,40%
2	100%	90,32%	100%	95,16%	97,40%
3	100%	100%	100%	98,39%	98,70%
4	93,33%	96,77%	100%	96,77%	100%
5	93,33%	100%	97,83%	98,39%	98,70%
Rata-rata nilai akurasi	95,99%	96,77%	98,69%	97,42%	98,44%

Dari tabel tersebut dapat digambarkan sebuah grafik hasil pengujian. Gambar 6.1 merupakan grafik hasil proses pengujian perbandingan rasio data latih dan data uji.



Gambar 6.1 Grafik Hasil Pengujian Perbandingan Rasio Data Latih dan Data Uji

Hasil pengujian yang di dapat ditampilkan pada Tabel 6.1 serta Gambar 6.1. Dapat dilihat bahwa hasil dari pengujian perbandingan rasio antara data latih serta data uji hanya sedikit berpengaruh terhadap setiap nilai akurasi. Nilai akurasi yang didapatkan setiap percobaan sudah baik, sebab *dataset* yang digunakan dalam pengujian nilai datanya sudah memiliki pola sendiri dan akurat. Hasil nilai akurasi yang terbaik yaitu menggunakan perbandingan rasio data sebesar 70% : 30% dimana nilai datanya yaitu 109 untuk data latih serta 46 untuk data uji sehingga mendapatkan nilai rata - rata dari rasio tersebut sebesar 98,69%. Untuk hasil dari pengujian perbandingan rasio data latih dan data uji menggunakan konsep lain didapat nilai rata-rata sebesar 85% (Cholissodin, 2016).

6.2 Pengujian dan Analisis Perbandingan Pengaruh Jumlah *Neuron Hidden Layer*

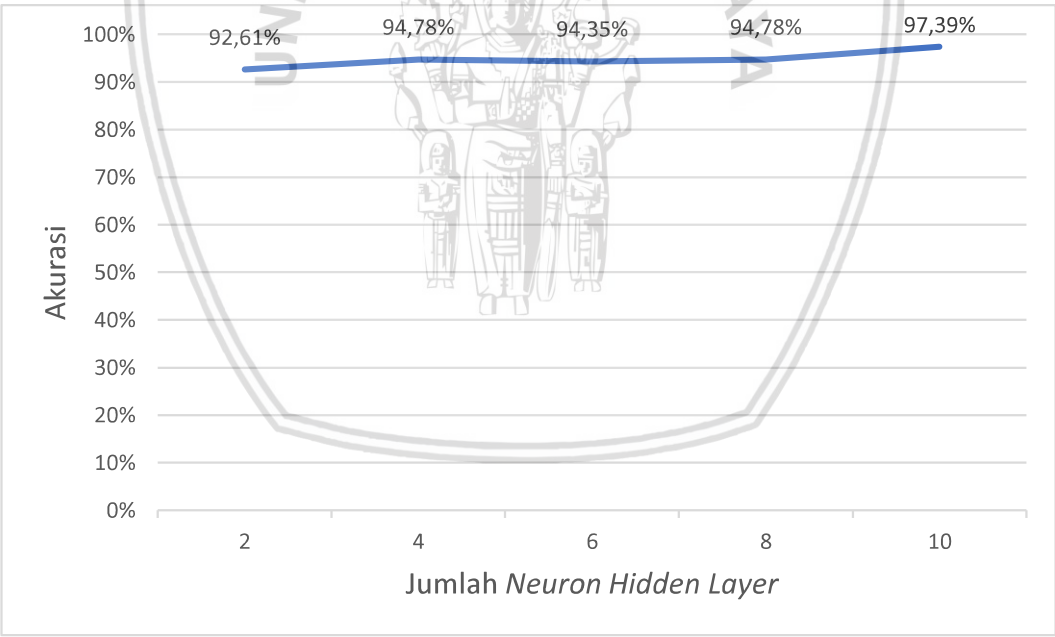
Pengujian untuk pengaruh jumlah nilai *neuron hidden layer* akan digunakan untuk mengetahui bagaimana pengaruh dari jumlah nilai *neuron hidden layer* terhadap hasil dari akurasi. Dalam pengujian ini nilai dari jumlah *neuron hidden layer* sudah ditentukan terlebih dahulu. Nilai jumlah *neuron hidden layer* yang akan digunakan untuk pengujian dan analisis yaitu sebesar 2, 4, 6, 8, 10. Inisialisasi parameter yang akan digunakan dalam pengujian ini adalah jumlah rasio antara data latih dan data uji sebesar 70% : 30% dimana nilainya dibulatkan menjadi 109 data latih dan 46 data uji, jumlah nilai *input neuron* (fitur) sebanyak 24 buah, dan menggunakan fungsi aktivasi sigmoid biner. Pengujian dan analisis pengaruh jumlah *neuron hidden layer* akan dilakukan sebanyak 5 kali percobaan. Pengujian ini akan dihasilkan nilai akurasi yang berbeda-beda. Semua ini terjadi karena nilai

dari bobot dan nilai bias yang diinisialisasikan secara *random*. Tabel 6.2 merupakan nilai hasil dari pengujian terhadap pengaruh jumlah *neuron hidden layer*.

Tabel 6.2 Hasil Pengujian Perbandingan Pengaruh Jumlah *Hidden Neuron*

Percobaan ke-i	2	4	6	8	10
1	95,65%	95,65%	91,30%	89,13%	100%
2	95,65%	93,48%	97,83%	93,48%	95,65%
3	93,48%	97,83%	95,65%	97,83%	97,83%
4	91,30%	97,83%	95,65%	95,65%	95,65%
5	86,96%	89,13%	91,30%	97,83%	97,83%
Rata-rata	92,61%	94,78%	94,35%	94,78%	97,39%

Dari tabel tersebut dapat digambarkan sebuah grafik hasil pengujian. Gambar 6.2 merupakan grafik hasil proses pengujian perbandingan pengaruh jumlah *neuron hidden layer*.



Gambar 6.2 Grafik Hasil Pengujian Perbandingan Jumlah *Hidden Neuron*

Berdasarkan hasil dari pengujian yang telah ditampilkan pada Tabel 6.2 serta Gambar 6.2 dapat dilihat bahwa hasil dari pengujian perbandingan jumlah *neuron hidden layer* cenderung memberikan hasil akurasi yang baik. Karena semakin banyak *neuron hidden layer* proses pelatihan akan lebih baik dan juga akan menghasilkan nilai akurasi yang baik pula. Hasil nilai akurasi terbaik yang dihasilkan pada pengaruh perbandingan jumlah *neuron hidden layer* yaitu menggunakan jumlah *neuron hidden layer* sebanyak 10 dimana nilai rata-rata dari

pengujian tersebut sebesar 97,39%. Untuk hasil dari pengujian perbandingan nilai neuron *hidden layer* menggunakan konsep lain didapat nilai rata-rata sebesar 90% (Cholissodin, 2016).

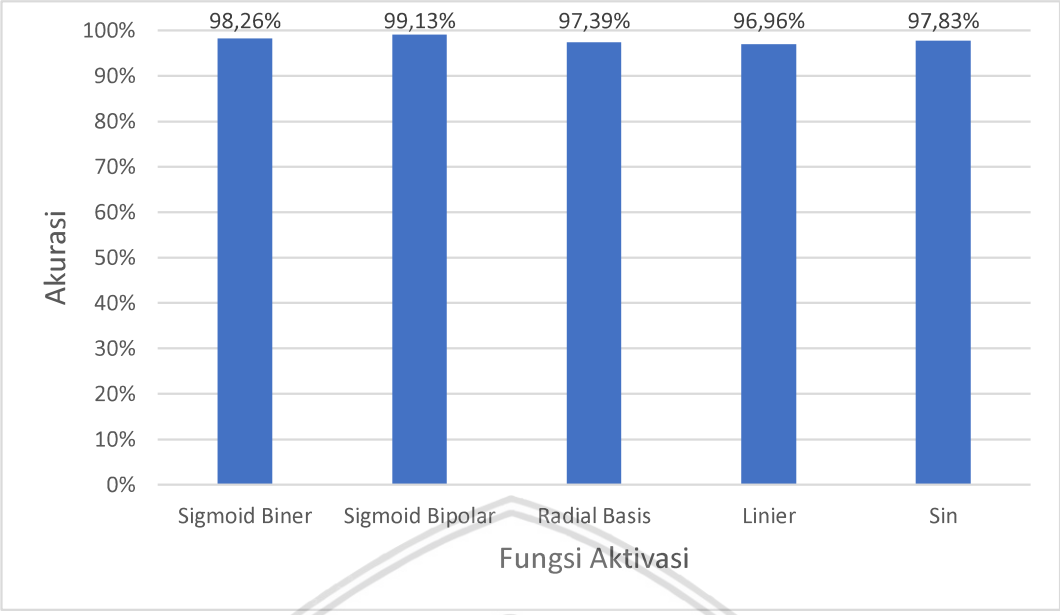
6.3 Pengujian dan Analisis Pengaruh Fungsi Aktivasi

Pengujian fungsi aktivasi bertujuan untuk mengetahui seberapa besar pengaruh dari fungsi aktivasi terhadap hasil dari akurasi. Dalam pengujian ini fungsi aktivasi sudah ditentukan terlebih dahulu. Fungsi aktivasi yang akan digunakan untuk pengujian dan analisis yaitu sigmoid biner, sigmoid bipolar, radial basis, linier, serta sin. Pengujian ini menggunakan parameter jumlah rasio antara data latih dan data uji sebesar 70% : 30% dimana nilainya dibulatkan menjadi 109 data latih dan 46 data uji, jumlah nilai *input neuron* (fitur) sebanyak 24 buah, dan nilai dari jumlah *neuron hidden layer* adalah 10. Pengujian dan analisis pengaruh fungsi aktivasi akan dilakukan sebanyak 5 kali percobaan. Pengujian ini akan dihasilkan nilai akurasi yang berbeda-beda. Semua ini terjadi karena nilai dari bobot dan nilai bias yang diinisialisasikan secara *random*. Tabel 6.3 merupakan nilai hasil pengujian fungsi aktivasi.

Tabel 6.3 Hasil Pengujian Pengaruh Fungsi Aktivasi

Percobaan ke-i	Sigmoid biner	Sigmoid bipolar	Radial basis	Linier	sin
1	97,83%	97,83%	95,65%	95,65%	95,65%
2	97,83%	97,83%	100%	97,83%	97,83%
3	97,83%	100%	95,65%	95,65%	97,83%
4	100%	100%	97,83%	100%	100%
5	97,83%	100%	97,83%	95,65%	97,83%
Rata-rata	98,26%	99,13	97,39%	96,96%	97,83%

Dari tabel tersebut dapat digambarkan sebuah grafik hasil pengujian. Gambar 6.3 merupakan grafik hasil proses pengujian pengaruh fungsi aktivasi.



Gambar 6.3 Grafik Hasil Pengujian Pengaruh Fungsi Aktivasi

Hasil pengujian yang di dapat ditampilkan pada Tabel 6.3 serta Gambar 6.3. Dapat dilihat hasil dari pengujian perbandingan fungsi aktivasi hanya berpengaruh sedikit terhadap setiap hasil akurasi. Fungsi aktivasi bipolar memiliki nilai rentang 1 sampai -1, ini menguntungkan untuk proses perhitungan ELM sebab nilai rentang 1 sampai -1 lebih berpengaruh dalam mendapatkan nilai *output weight*. Hasil akurasi terbaik yang dihasilkan pada pengaruh perbandingan fungsi aktivasi yaitu menggunakan fungsi aktivasi sigmoid bipolar dengan didapatkan nilai rata-rata dari setiap percobaan sebesar 99,13%. Untuk hasil dari pengujian perbandingan pengaruh fungsi aktivasi menggunakan konsep lain didapat nilai rata-rata sebesar 85% (Cholissodin, 2016).

BAB 7 PENUTUP

Pada bagian penutup membahas mengenai kesimpulan dari penelitian yang sudah dilakukan serta membahas mengenai saran terhadap penelitian yang sejenis ataupun pengembangan penelitian yang sedang dilakukan ini.

7.1 Kesimpulan

Berdasarkan proses pengujian serta hasil dari proses analisis mengenai klasifikasi gagal ginjal kronis menggunakan *Extreme Learning Machine* didapatkan kesimpulan bahwa:

1. Dapat dilihat dari nilai proses pengujian, bahwa algoritme *Extreme Learning Machine* bisa digunakan untuk klasifikasi *dataset* gagal ginjal kronis dengan nilai akurasi yang terbaik didapat pada rasio antara data latih serta data uji 70% : 30% yang mana perbandingan banyak datanya sebesar 109:46, nilai *neuron hidden layer* berjumlah 10, serta menggunakan sigmoid bipolar.
2. Dari penggunaan ketiga pengujian yaitu pengujian rasio data latih dan data uji, pengujian jumlah *neuron hidden layer* serta pengujian penggunaan fungsi aktivasi untuk klasifikasi risiko gagal ginjal kronis dihasilkan nilai akurasi yang terbaik sebesar 99,13%.

7.2 Saran

Saran mengenai kelanjutan penelitian dalam klasifikasi gagal ginjal kronis menggunakan algoritme *Extreme Learning Machine* yaitu sebagai berikut:

1. Untuk penelitian selanjutnya diharapkan untuk melakukan optimasi pada penggunaan nilai *weight* serta bias supaya nilai *weight* serta bias yang di dapat lebih optimal, karena setiap kasus sangat besar kemungkinan memiliki nilai *weight* serta bias yang berbeda untuk mendapatkan hasil pengujian yang optimal.
2. Untuk penelitian selanjutnya diharapkan untuk mengatasi masalah dengan data yang *outlier* karena data *outlier* akan menimbulkan nilai akurasi menjadi tidak optimal.

DAFTAR PUSTAKA

Andriyani, R., Triana, A. dan Juliarti, W. 2015. Buku Ajar Biologi Reproduksi dan Perkembangan. Sleman: *Deepublish*.

Candra, A., 2013. Rajin Pantau Tensi Turut Sehatkan Ginjal. Tersedia di: < http://health.kompas.com/read/2013/03/06/18435262/rajin.pantau.tensi.turut.sehatkan.ginjal> [Diakses 28 Februari 2018].

Cholissodin, I., Sutrisno, Soebroto, A. A., Hanum, L., Caesar, C. A. 2017. Optimasi Kandungan Gizi Susu Kambing Peranakan Etawa (PE) Menggunakan ELM-PSO di UPT Pembibitan Ternak dan Hijauan Makanan Ternak Singosari Malang.

Hosten, A.O., 1990. BUN and Creatinine. In: Walker, H.K., Hall, W.D., Hurst, J.W., 3rd eds. *Clinical Methods: The History, Physical, and Laboratory Examinations*. Boston: Butterworths. p. 874–8.

Huang, G.B., Zhu, Q.Y. and Siew, C.K., 2006. *Extreme learning machine: theory and applications. Neurocomputing*, 70(1), pp.489-501.

Humaini, Q. 2015. Jaringan Syaraf Tiruan *Extreme Learning Machine* (ELM) Untuk Memprediksi Kondisi Cuaca di Wilayah Malang. Malang.

Imanurrahman, M. 2015. Pengklasifikasian Penyakit Ginjal Menggunakan Metode *Naive Bayes.* Universitas Sumatra Utara.

Jain, Y.K. and Bhandare, S.K., 2011. *Min max normalization based data perturbation method for privacy protection*. *International Journal of Computer & Communication Technology*, 2(8), pp.45-50.

Kusuma, S. 2003. *Artifical Intelligence* (Teknik dan Aplikasinya). Yogyakarta: Graha Ilmu.

Munandar, M. 2016. Faktor-Faktor Risiko Yang Berhubungan Dengan Kejadian Gagal Ginjal Kronis Pada Pasien Rawat Jalan di RSUD Dr Moewardi Surakarta. Surakarta.

National Kidney Foundation. 2002. K/DOQI *Clinical Practice Guidelines for Chronic Kidney Disease*: *Evaluation, Clasification and Stratification.*

Perdana, J. A., Soeprijanto, A., Wibowo, R. S. 2012. Peramalan Beban Listrik Jangka Pendek Menggunakan *Optimally Pruned Extreme Learning Machine* (OPELM) pada Sistem Kelistrikan Jawa Timur. Surabaya.

Pranandari, R dan Supadmi, W. 2015. *Risk Factors Cronic Renal Failure on Hemodialysis Unit in* RSUD Wates Kulon Progo. Wates Kulon Progo.

Puspitaningrum, D. 2006. Pengantar Jaringan Saraf Tiruan. Yogyakarta: Andi.

Shahsavari, M.K., Bakhsh, H.R. & Rashidi, H., 2016. *Efficient Classification of Parkinson's Disease Using Extreme Learning Machine and Hybrid Particle Swarm Optimization*. 2016 4th *International Conference on Control*,

Instrumentation, and Automation (ICCIA). Qazvin Islamic Azad University. Qazvin, Iran, 27-28 January 2016.

Simanjuntak, T. H., Mahmudy, W. F. & Sutrisno, 2017. Implementasi Modified *K-Nearest Neighbor* Dengan Otomatisasi Nilai K Pada Pengklasifikasian Penyakit Tanaman Kedelai. Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, Volume 1, pp. 75-79.

Siwi, Iga Permata, Imam Cholissodin, and M. Tanzil Furqon, 2016. Peramalan Produksi Gula Pasir Menggunakan *Extreme Learning Machine* (Elm) Pada Pg Candi Baru Sidoarjo.

Srimuang, W., & Intarasothonchun, S., 2015. *Classification Model of Network Intrusion using Weighted Extreme Learning Machine*. 12th *International Joint Conference on Computer Science and Software Engineering* (JCSSE), 190-194.

The Renal Association. 2013. CKD Stages. Tersedia di: <http://www.renal.org/information-resources/the-uk-eckd-guide/ckdstages#sthash.frm4MEB8.dpbs>. [Diakses pada 28 Februari 2018].

Wilson, L. M. 2005. Gagal Ginjal Kronik. Dalam: Wilson, L.M., Price, S.A., penyunting. Patofisiologi: konsep klinis proses-proses penyakit. Edisi ke-6. Jakarta:ECG. hlm. 912–47.

Yaseen, Z. M., Deo, R. C., Hilal, A., Abd, A. M., Bueno, L. C. B., Sanz, S. S. & Nehdi, M. L. 2017. *Predicting Compressive Strength Of Lightweight Foamed Concrete Using Extreme Learning Machine Model*.